(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(51) Int. Cl.[7]
G01R 31/319

(11) 공개번호 특2001-0013719
(43) 공개일자 2001년02월26일

(21) 출원번호 10-1999-7011734
(22) 출원일자 1999년12월11일
번역문제출일자 1999년12월11일
(86) 국제출원번호 PCT/US1998/11557
(87) 국제공개번호 WO 1998/57187
(86) 국제출원출원일자 1998년06월04일
(87) 국제공개일자 1998년12월17일
(81) 지정국 EP 유럽특허 : 오스트리아 벨기에 스위스 독일 덴마크 스페인 프랑스 영국 그리스 아일랜드 이탈리아 룩셈부르크 모나코 네덜란드 포르투칼 스웨덴 핀랜드 사이프러스
국내특허 : 일본 대한민국 싱가포르
(30) 우선권주장 08/874,615 1997년06월13일 미국(US)
(71) 출원인 테라다인 인코퍼레이티드 레카 도날드 지.
미국 매사츄세츠 02118 보스톤 해리슨 애비뉴 321
(72) 발명자 프로스카우어다니엘씨.
미국매사츄세츠02165뉴튼더비스트리트240
데시판데프라딥비.
미국매사츄세츠01803벌링톤비컨스트리트26아파트먼트121디
(74) 대리인 장용식

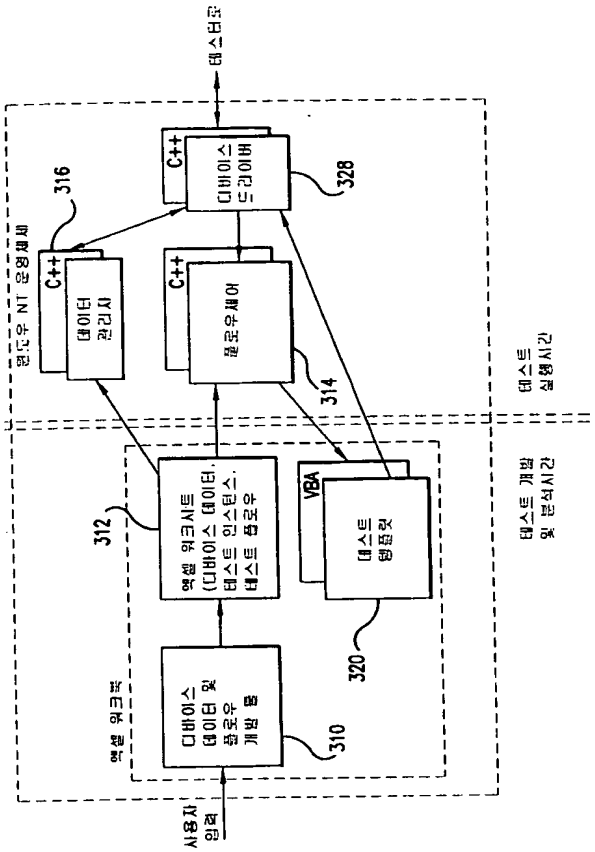
심사청구 : 없음

(54) 저비용으로 용이하게 사용할 수 있는 자동 테스트 시스템소프트웨어

요약

저비용으로 용이하게 테스트 프로그램을 개발하고 실행하는 소프트웨어로 반도체 디바이스를 자동으로 테스트하는 장치. 테스터는 상업적으로 이용가능한 스프레드시트 프로그램을 실행하는 컴퓨터 워크스테이션으로 제어된다. 상업적으로 이용가능한 스프레드시트 프로그램은 프로그램 개발 환경을 제공하기 위해 애플리케이션에 세팅된다. 부가하여, 상업적으로 이용가능한 스프레드시트 프로그램으로 만들어진 프로그램은 반도체 디바이스상의 테스트 실행을 제어한다. 테스터는 상업적으로 이용가능한 스프레드시트 프로그램의 사용이 공지된 프로그래밍 인터페이스를 발생시키기 때문에 쉽게 프로그래밍된다. 이러한 방식으로, 상업적으로 이용가능한 스프레드시트 프로그램은 애플리케이션에 의해 사용되는 스프레드시트 기능을 단지 제공한다기 보다는 오히려 테스터를 제어하는 소프트웨어를 구현한다. 자동 테스트 장치를 제어하는 소프트웨어는 따라서 매우 쉽게 프로그래밍되거나 또는 수정된다. 또한 프로그래밍하기가 매우 쉽다.

대표도



색인어

워크스테이션, 비쥬얼 베이직, 엑셀, 객체 지향 프로그램, 집적 프로그래밍 언어

명세서

기술분야

본 발명은 일반적으로 반도체 제조에 사용되는 자동테스트 장치에 관한 것이고, 더 특별하게는 테스트 시스템에 대한 소프트웨어 제어에 관한 것이다.

배경기술

자동테스트 장치는 반도체 제조에 폭넓게 사용된다. 반도체는 일반적으로 제조동안에 적어도 한번은 테스트되고, 때때로 제조공정에서 적어도 한단계이상 테스트된다. 모든 소자가 테스트되기 때문에, 테스트의 속도는 반도체의 경제적인 제조에 중요하다. 속도가 더딘 테스트는 반도체 소자를 만드는데 필요한 비싼 자본설비의 충분한 이용을 막는다.

최신의 반도체 소자는 매우 복잡하고 많은 동작상태를 갖는다. 각각의 이들 동작상태는 완벽한 테스트를 실행하도록 작용되어야 한다. 따라서, 자동테스트 장치는 테스트 데이터를 반도체 디바이스에 적용하고, 제조세팅에서 매우 빨리 많은 측정을 하도록 설계된다. 도 1은 종래기술의 일반화된 자동테스트 시스템을 도시한다. 철저하고 빠른 테스트를 제공하기 위해서, 자동테스트 시스템은 일반적으로 테스터 몸체(112), 컴퓨터 워크스테이션(110), 및 핸들링 디바이스(114)를 포함한다.

컴퓨터 워크스테이션(110)은 핸들링 디바이스(114) 및 테스터 몸체(112)를 제어한다. 이것은 테스터 몸체(112)상의 복수의 테스트 프로브(118)와 접촉하는 곳에 반도체 디바이스(도시생략)가 위치하도록 핸들링 디바이스(114)를 제어한다. 종종, 테스터는 테스트 프로브(118)를 포함한 개별 테스트 헤드를 포함한다. 그러나 이러한 특징은 본 발명에서는 중요하지 않다.

다음, 워크스테이션(110)은 테스트중인 디바이스상에 일련의 테스트를 실행하도록 테스터 몸체(112)를 제어한다. 각 테스트는 일반적으로 제어신호가 워크스테이션(110)으로부터 테스터 몸체(112)로 송신되는 세트업 부분을 포함한다. 제어 신호는 항상 버스(116)를 통해 송신되는 디지털 값이다. 이들 제어신호는 테스트를 필요로 하는 측정을 하도록 하드웨어를 테스터 몸체(112)내에 배치시킨다. 테스터 몸체내의 하드웨어는 자극을 제공하고, 제어신호에 따라 테스트중인 디바이스로부터 응답을 측정한다.

도 1은 테스터 몸체(112)내의 하드웨어가 핀(124)으로서 구별되는 복수의 회로를 포함한다. 각 핀(124)

은 신호를 발생시키거나, 테스트 프로브(118)중의 하나에 대해 측정을 한다. 각 핀은 정적, 또는 DC 신호를 측정하거나 제공한다. 대안적으로, 각 핀(124)은 때때로 "버스트(burst)"로 불리우는 데이터 변경을 측정하거나 제공한다.

버스트동안 테스터 몸체(112)는 타이밍 및 시퀀스 회로(120)에 의해 제어된다. 타이밍 및 시퀀스 회로(120)는 각각의 핀(124)이 시퀀스의 데이터 값을 연관메모리(128)로부터 판독하도록 하게 한다. 각 데어터 값은 핀이 제시간에 특정점에서 연관된 테스트 프로브(118)에서의 측정을 적용하거나 예상하는 신호의 형태를 지시한다.

모든 핀(124) 값을 정어하는 세트의 데이터 값은 "벡터"로 불리는 시간의 측정을 예상하거나 제공한다. 벡터는 테스트중인 디바이스의 실제 동작 조건을 시뮬레이션하도록 매우 고속으로 실행되어야 한다. 보통, 반도체 디바이스 테스트에 필요한 버스트를 정의하는 수백만개의 벡터가 있다. 테스트 시스템이 특정형태의 부분을 테스트하도록 프로그래밍되어지는 순간에 벡터는 전형적으로 메모리 (128)로 로딩된다. 이러한 로딩 프로세스는 수분이 걸리고, 각각의 버스트에 대해 반복하지 않는다. 다소, 각 버스트에 대해, 워크스테이션(110)은 벡터가 버스트의 파트로서 실행하도록 지시하는 커맨드를 송신한다. 일단 버스트가 완성되면, 워크스테이션(110)은 메모리(128) 또는 타이밍 및 시퀀스 회로(120)로부터 버스트의 결과를 판독한다.

부가적으로, 테스터 몸체(112)는 하나 이상의 장치(126)를 포함한다. 장치는 특정 테스트기능을 수행한다. 예를 들면, 이것은 사인파와 같은 특정 테스트신호를 생성한다. 대안적으로, 장치는 고속으로 신호를 샘플링하여, 디지털 신호 프로세서에의해 뒤에 분석될 수 있다. 이러한 기능은 버스트의 파트로서 수행되어지거나 버스트로부터 분리되어 수행될 수 있다.

때때로 "작업(job)"으로 불리는 한 파트의 모든 테스트는 장치(126)에 의한 측정 또는 DC측정으로 산재되어 있는 일련의 버스트로 구성된다. 버스트는 테스트중인 디바이스의 특정 기능의 속성을 측정하도록 사용된다. 대안적으로, 각 버스트는 DC측정이 될 수 있는 상태로 측정되어야 할 디바이스를 위치시키는데 단지 사용될 수 있다. 때때로 "플로(flow)"로 불리는, 테스트의 요소가 수행되는 순서는 워크스테이션(110)의 소프트웨어에의해 지시되어진다.

일단 디바이스가 전부 테스트되거나, 결함이 있다고 판단되는 점에 테스트되면, 워크스테이션(110)은 핸들링 디바이스(114)로 제어신호를 생성시킨다. 다음, 핸들링 디바이스(114)는 테스터 몸체(112)를 테스트되어야 할 다음 디바이스에 제공하고, 이러한 프로세스가 반복된다. 또한 워크스테이션(110)은 특정 디바이스가 합격인지 불합격인지에 대한 데이터를 모은다. 이것은 데이터를 프로세스할 수 있어 결함이 있는 디바이스는 버려지나, 또는 실행경향에 대해 데이터를 분석하는 등의 다른 기능을 수행할 수 있다.

워크스테이션(110)내의 또 다른 중요한 기능은 특정 디바이스에 대한 또는 디바이스 자체 설계동안 테스트의 개발을 돕는다는 것이다. 개발 동안, 테스트 엔지니어는 테스터 몸체(112)내의 하드웨어에 대해 세팅하는 것과 같은 일을 특정할 수 있어야 한다. 엔지니어는 또한 테스트의 플로를 특정할 수 있어야 한다. 테스터를 제어하는 소프트웨어는 익히거나 사용하기에 매우 쉽고 다루기 쉬운 것이 가장 바람직하다. 반도체 시장은 매우 경쟁적이고, 기술혁신에의해 큰부분으로 나아간다. 따라서, 칩이 가능한한 빨리 제조를 신속하도록 만들어지고 개발되는 것이 중요하다. 만약 지연이 생긴다면 테스트 엔지니어는 어떻게 테스터를 프로그래밍하는지 익힐 필요가 있기 때문에 바람직하다 못하다.

도 2는 워크스테이션(110)내의 소프트웨어에 대한 전통전인 소프트웨어 구조를 도시한다. 소프트웨어는 테스트 개발 및 분석시간에 사용되는 요소와, 테스트 실행시간에 사용되는 소프트웨어로 분할된다.

테스트를 정의하는 제 1 단계는 일반적으로 디바이스 데이터를 특정하는 것이다. 도 2는 디바이스 데이터 개발 툴(210)을 도시한다. 전형적으로, 이들 툴은 테스터에 대해 주문개발된 데이터베이스에 데이터를 저장하고 테스트되도록 특정 디바이스에 관한 데이터를 갖는다. 이 정보는 핀 그룹 및 시간 세트와 같은 것으로 정의한다. 반도체 디바이스는 복수의 디바이스 리드선을 갖는다. 종종, 이들 리드선은 그룹으로 함께 동작한다. 예를 들면, 그룹은 주소 버스나 데이터 버스를 대표한다. 따라서, 한 툴은 리드선의 그룹이 특정되도록 하게 한다. 다른 툴은 테스터상의 핀 및 디바이스상의 특정 리드선 사이의 상관관계가 특정되도록 하게 한다. 일부 경우에서, 특정 데이터피스는 다른 데이터 값으로 평가하는 방정식 관계에 의해 정의된다. 그러나, 종래기술에서, 방정식 핸들링 능력은 제한되어졌다.

다른 툴은 신호의 타이밍이 특정되도록 하게 한다. 각 핀(124)은 타이밍 "에지"와 관련되어 정의된 신호를 측정하거나 생성할 수 있다. 각 핀에 대해 1 세트의 타이밍 에지가 있다. 각 에지는 실행된 각 벡터에 대해 한번 발생할 수 있다. 각 에지가 발생된 시간은 프로그래밍될 수 있다. 각 핀에 대해 각 에지에 대한 프로그래밍된 시간은 1 세트의 타이밍 데이터를 대표한다. 바람직한 프로그래밍은 다른 버스트에 또는 심지어 동일 버스트내에도 다르기 때문에, 다중 시간세트는 보통 특정되게 된다. 각 시간세트는 입력되고 저장되야 한다. 이 정보는 전형적으로 특정 테스터에 대해 설계된 커스텀 데이터 베이스 프로그램을 통해 입력된다. 커스텀 데이터 베이스를 통해 입력된 정보는 디바이스 데이터 파일(214)내에 저장된다.

진입된 정보의 다른 형태는 1 테스트에 대해 수행되야 되는 특정 단계이다. 이 정보는 테스트 템플릿(220)에서 특정된다. 템플릿은 임의의 특정 디바이스 데이터 파일의 값과 무관하게 설명되는 1 세트의 단계이다. 테스트 템플릿은 특정 테스트 시스템에 대해 정의되는 포맷으로 데이터 파일로서 기록된다. 이들 파일은 일반적으로 테스터 제조자에의해 공급되고, 전형적으로 고속 실행으로 잘 알려져있는 C 프로그래밍 언어와 같은 프로그래밍 언어로 기록된다. 일부 경우에서, 테스트 엔지니어는 테스트 템플릿을 주문할 것이다.

테스트 템플릿은 컴파일러(250)에 의해 컴파일된다. 컴파일된 프로그램이, 번역된 프로그램보다 더 빨리 실행된다는 것은 의례적으로 인식되어져 있다. 테스터에서의 요구되는 실행속도 때문에, 컴파일된 테스트 프로그램이 바람직하다.

커스텀 플로 툴(218)은 테스트의 순서가 특정되도록 하게 한다. 플로는 프로그램 플로(222)로서 기록된

다. 전형적으로, 플로는 테이블에서의 일련의 엔트리로서 특정되지만, 전통적인 프로그래밍 언어를 닮은 특정 프로그래밍 언어에서도 될 수 있다.

플로의 단계를 특정하기 위해, 특정 테스트 템플릿은 그 단계를 위해 사용되는 디바이스 데이터 파일과 함께 특정될 수 있다. 일부의 템플릿은 실행되도록 버스트를 특정할 것이다. 다른 템플릿은 테스트의 결과가 테스터(112) 안쪽의 하드웨어로부터 판독되거나, 하드웨어가 특정 방법으로 세트업되도록 특정한다. 플로에서의 다른 단계는 전통적인 프로그래밍 언어에서 발견되는 브랜칭 및 루핑 명령어와 같은 제어 스테이트먼트이다.

프로그램 플로는 입력으로서 커스텀 실행(224)에 제공한다. 커스텀 이그제큐티브(224)는 디바이스 데이터 파일(214)로부터의 데이터로 프로그램 플로에서 특정된 템플릿에 채워진다. 이것은 테스트 오브젝트를 만든다.

테스트 시간에서, 커스텀 이그제큐티브(224)는 테스트 오브젝트를 프로세스하고 실제적으로 테스터(112)를 제어한다. 커스텀 이그제큐티브(224)는 디바이스 드라이버(228)을 통해서 데이터 및 커맨드를 테스터(112)로 송신한다. 커스텀 이그제큐티브(224)가 동작하는 속도는 전체 테스트 시간을 지시하는데 큰 역활을 하기 때문에 커스텀 실행는 전통적으로 C-언어와 같은 프로그래밍 언어로 기록되어, 그 결과 프로그래머에게 특정하는 많은 옵션을 준고, 따라서 이것의 실행을 최적화한다.

디바이스 드라이버(228)는 테스터(112)내측에 하드웨어를 제어하도록 버스(116)를 통해 정보를 송수신하는 프로그램 수속이다. 디바이스 드라이버(228)는 저레벨 하드웨어 제어를 수행하기 때문에, C 언어와 같은, 상기 제어를 하게 하는 프로그래밍 언어로 기록된다.

모든 디바이스는 가능한한 빨리 테스트되는 것이 바람직하다. 따라서, 프로그램 플로에서의 각 단계를 실행하는데 필요한 시간의 양이 매우 짧은 것이 바람직하다. 테스터(112) 또는 핸들링 디바이스(114)가 커맨드를 "실행" 또는 응답하는데 걸리는 실제 시간은 워크스테이션(110)내의 소프트웨어와 무관하다. 그러나, 워크스테이션내(110)의 소프트웨어는 "오버헤드" 또는 각 커멘드를 세트업하는데 걸리는 시간의 양을 결정한다. 작업의 실행시간을 매우 짧게 하는데 상당한 수고가 소비된다. 따라서, 오버헤드 타임이 매우 짧게 되는 것이 매우 중요하다. 종래 기술의 시스템에서, 이것은 커스텀 이그제큐티브(224)로 성취되었다. 커스텀 이그제큐티브(224)는 전통적으로 C 언어로 기록된다. C 언어는 빨리 동작하도록 설계될 수 있는 이그제큐티브 개발을 할 수 있는 것으로 폭넓게 인식받는다.

C 언어로 커스텀 이그제큐티브(224)를 준비하는데 생기는 주요 문제는 상기 이그제큐티브를 기록하는데 매우 오랜 시간이 걸린다는 것이다. 비용도 수 백만 달러가 들고 수많은 수고가 든다. 커스텀 이그제큐티브를 갖는 또 다른 문제는 사용자 인터페이스가 또한 커스텀 인터페이스라는 것이다. 이것은 테스터를 동작하는 엔지니어가 테스터를 프로그래밍을 어떻게 하는지 익히는 데 시간을 투자해야 함을 의미하며, 이러한 것은 다수의 사용자에게 바람직하지 못하다. 그러나, 지금까지, 이들 단점은 테스터를 빨리 작동할 수 있게 하기 위해 필요한 것을 인식되어 왔다.

C 언어에서의 커스텀 이그제큐티브의 다른 결점은 매우 제한된 세트의 사용자 인터페이스 능력을 가진다는 것이다. 사용자 인터페이스는 테스터에 대해 특히 중요하고 다른시간에서 매우 다른 기술 레벨의 사람들에의해 동작된다. 디바이스 및 테스트 개발동안, 테스터는 고도로 숙련받은 엔지니어에의해 동작된다. 그러나, 일단 디바이스 및 테스트가 개발되면, 테스터는 반숙련의 제조 작업자에의해 동작된다. 따라서, 사용자 인터페이스는 제조 세팅에서 매우 단순화되어야 한다.

테스터는 전통적으로 워크스테이션(110)에서 구동되는 개별 제조 인터페이스 소프트웨어를 가진다. 사용자 인터페이스는 전통전으로 메뉴로서 구현된다. 테스트 엔지니어는 동작자가 제조 테스트 동안 필요한 이들 커맨드를 단지 메뉴상에 위치시킨다. 테스터에 제공된 소프트웨어는 따라서 메뉴의 형태로 사용자 인터페이스를 만들도록 능력이 제공된다. 불행히도, 메뉴는 계층형이지만 많은 동작은 그러지 못하다. 따라서, 비계층형의 인터페이스가 빠르고 쉽게 개발되도록 하는 소프트웨어를 갖는 테스터를 제공하는 것이 바람직하다.

도 2는 워크스테이션(110)내에 포함된 다른 소프트웨어 요소를 도시한다. 메모리(128)(도 1)에 로딩된 벡터는 많은 양의 데이터를 대표한다. 이들은 항상 소프트웨어 시뮬레이터에의해 개발되거나 자동화된 방법으로 파생된다. 이들은 사용자에의해 항상 수동으로 입력되는 것은 아니다. 도 2는 벡터 개발툴이 시뮬레이션 입력을 수신하는 것을 도시한다. 벡터 개발툴은 사용자가 벡터 파일을 편집하게 하고, 다음 벡터 및 결과 파일(234)에 저장된다. 벡터 파일은 테스트이전에 메모리 (128)로 로딩된다.

마찬가지로, 예를 들어, 복수의 연속 사이클동안 테스트중인 디바이스에서 예시적인 측정값을 대표하는 많은 양의 데이터가 테스트동안 얻어지는 곳에서, 버스(116)(도 1)를 통해 다시 통과된다.

벡터 파일(234)은 또한 디버그 및 분석툴(230)에 의해서 액세스된다. 디버그 및 분석툴은 테스트를 개발하거나 테스트된 파트를 개발하는 엔지니어에 의해서 사용된다. 그래픽적으로 결과 및 벡터를 디스플레이하는 것등을 하는 이들 툴은 사용자가 벡터 파일을 변조하거나 측정값 위에 기대값을 슈퍼임포즈하도록 하게 한다. 추가로, 디버그 및 분석툴은 사용자가 테스터(112)에서의 하드웨어를 직접 프로그래밍하게 한다.

전형적인 시나리오에서, 디바이스 데이터 및 개발툴(212), 테스트 및 플로 개발툴(218), 디버그 및 분석툴, 및 테스트 벡터 개발툴(232)이 테스터에 공급된다. 테스트된 디바이스가 여전히 개발중인 동안, 사용자는 디바이스 데이터 및 개발 툴(212)을 디바이스의 특성을 특정하도록 사용한다. 테스트 및 플로 개발툴 (218)은 또한 디바이스를 구동하도록 일부 테스트를 특정하는데 사용될 수 있다.

샘플 디바이스가 제조된다. 테스트 프로그램이 구동되고, 디버그 및 분석툴이 샘플 디바이스를 체크아웃하도록 사용된다. 프로세스에 있어서, 샘플 디바이스의 결점이 검출되거나 프로그램에서의 문제가 검출된다. 이들 문제는 수정되고 추가 테스트가 생성되고 구동된다.

개발 프로세스의 목적에서, 반도체 디바이스 및 테스트 프로그램이 제조로 전이될 것이다. 제조에서, 각

파트는 이것이 제조될 때 테스트된다.

발명의 개요

전술한 배경기술을 상기하면, 본발명의 목적은 빠르고 저렴하게 개발될 수 있는 테스터를 제어하는 소프트웨어를 제공하는 것이다.

또한 사용자가 사용하는데 빠르고 쉽게 익힐 수 있는 테스터를 위한 소프트웨어를 제공하는데 있다.

앞에서 및 다른 목적은 데이터 파일, 테스트 템플릿, 및 프로그램 플로를 개발하는 상업적으로 이용가능한 스프레드시트 소프트웨어를 사용한 테스터에서 성취된다. 우리는 매우 저렴한 비용에서 요구된 속도로 모든 기능을 제공하고, 추가로 비싼 숙련비없이 사용자가 사용을 익힐 수 있는 표준 사용자 인터페이스가 있는 이점을 제공하는 상업적으로 이용가능한 스프레드시트 소프트웨어를 알아냈다.

도면의 간단한 설명

본 발명은 다음에 오는 상세한 설명 및 첨부된 도면에 의해서 더 잘 이해가 될 것이다.

도 1은 종래기술의 테스터의 하드웨어 블록도,

도 2는 종래기술 테스터에 대한 소프트웨어의 소프트웨어 블록도,

도 3은 본 발명에 따른 소프트웨어의 소프트웨어 블록도,

도 4a 내지 9는 본 발명을 사용한 디바이스 데이터 시트의 설명도,

도 10a 내지 11c는 본 발명을 사용한 플로 데이터 시트의 설명도,

도 12a 및 12b는 샘플 테스트 템플릿을 도시,

도 13은 본 발명에 따른 플로 제어 소프트웨어의 설명도,

도 14는 본 발명에 따라 쉽게 구현될 수 있는 옵션 프로그래밍 인터페이스,

도 15는 대안적인 상업적으로 이용가능한 스프레드 시트 프로그램을 사용하여 구현된 디바이스 데이터 시트의 예시도,

도 16a 및 16b는 옵션 디바이스 데이터 및 플로 개발툴의 예시도,

발명의 상세한 설명

도 3은 종래기술 테스터에서의 커스텀 소프트웨어의 대부분의 기능이 상업적으로 이용가능한 스프레드시트 프로그램으로 수행될 수 있는 우리가 알고 있는 유일한 소프트웨어 구조를 도시한다. 바람직한 실시예에서, 워싱턴 레드몬드의 마이크로소프트사 제품인 EXCEL이 사용된다.

EXCEL은 사용자가 "워크북"을 만들수 있게 하는 스프레드시트 프로그램이다. 워크북은 상이한 스프레드시트들로 구성된 데이터의 집합체이다. 스프레드시트는 단지 그리드 형태의 셀들이며, 각 셀은 데이터 값이 다른 셀에 있는 값들로부터 어떻게 계산되는지를 정의하는 등식 또는 데이터의 조각을 유지한다. 상기 셀들은 순차적인 그리드형태로 나타난다. 셀들은 일반적으로 관계된 데이터가 특정한 행 또는 특정한 열에 있는 셀에 나타나도록 구성된다. 각 셀들상의 동작용 도구는 데이터가 무엇을 나타내는지에 관계없이 동일하다. 워크북내에서, 임의의 스프레드시트가 액세스될 수 있고 임의의 스프레드시트에 있는 데이터가 다른 스프레드시트에서 사용될 수 있다.

EXCEL은 또한 스프레드시트를 정의하고 "애플리케이션"을 개발하는 특징을 포함한다. EXCEL은 회계 시스템같은 애플리케이션에 사용되어 왔다. 스프레드시트의 형태는 특정되어 사용자는 단지 특정 일에 관계된 셀을 볼 것이다. 디스플레이는 EXCEL 애플리케이션 개발자 가이드와 1996년 Que Corporation이 발행한 ISBN 0-7897-0269-X인 Jeff Webb의 "애플리케이션을 위한 엑셀 비쥬얼 베이직 사용"에 설명된 다른 방법으로 만들수있다. 상기 각각은 여기에서 참조로써 사용된다.

EXCEL이 애플리케이션을 제작할 수 있도록 사용될 수 있는 두번째 방법은 비쥬얼 베이직의 사용을 통해서 가능하다. 비쥬얼 베이직은 EXCEL로 구축된 프로그래밍 언어이다. 이름에서 알 수 있는 바와 같이, 비쥬얼 베이직의 제 1 목적은 사용자가 데이터를 입력하여 스프레드시트가 표현되는 것에 따라 스프레드시트의 형태를 제어하는 것이다. 비쥬얼 베이직은 일반적으로 애플리케이션 프로그램이 수행되는 동안 임의의 소정 시간에 사용자에게 표현되는 스프레드시트를 선택하는데 사용된다. 선택은 일반적으로 사용자가 입력한 커맨드 또는 데이터에 근거하며, 임의의 소정 작업이 진행되는 동안, 사용자는 상기 작업과 관계한 데이터를 본다. 따라서, 비쥬얼 베이직은 진행중인 애플리케이션이 사용자에게 나타나는 방법을 제어하기 위해 사용될 수 있는 많은 커맨드를 포함한다. 비쥬얼 베이직은 또한 브랜칭 (branching)과 루핑(looping)에 대한 조건상태와 같은 프로그램 흐름을 제어하는 프로그래밍 커맨드를 포함한다.

비쥬얼 베이직이 반도체 테스터의 워크스테이션에 사용될 때 테스트의 실행을 제어하는 것뿐만 아니라 테스트 템플릿을 기록하도록 사용될 수 있다는 것을 우리는 발견했다. 특히, Excel같은 상업적 스프레드시트 프로그램은 너무 빨라서 주문 실행에 비하여 테스트를 실행하는 속도의 다소의 손실이 없도록 데이터와 테스트 템플릿으로부터 테스트 프로그램을 조립하여 주문제작할 수 있다는 것을 우리는 발견했다. 상기와 같이 이해하여, 우리는 많은 이점을 이룰 수 있다. 그러한 소프트웨어 애플리케이션의 개발은 매우 빠르다. 부가하여, 대부분의 프로그래머들은 상업적 스프레드시트 프로그램과 친숙하기 때문에 일반적으로 상기 애플리케이션 사용방법을 배우는 것은 매우 쉽다. 더우기, 테스터 제어 소프트웨어는 문맥의존 도움과 편집 커맨드와 같은 주문형 스프레드시트로 구축된 많은 특징을 갖는 이점이 있다.

도 3은 디바이스 데이터와 플로우 개발 툴(310)을 도시한다. 바람직한 실시예에서, 상기 툴은 Excel 워

크북내에서 스프레드시트를 커스터마이징함으로써 구현된다. 디바이스 데이터를 유지하기 위해, 핀 맵, 각 핀에 인가될 신호의 레벨 및 타임 세트에 대한 스프레드시트가 있다. 선택적으로 채널 맵, 다양한 상수값의 명세와 에지 세트와 같은 데이터에 대한 다른 스프레드시트가 있을 수 있다. 다양한 테스트에 대하여 상이한 데이터값이 요구되기 때문에 각 타입의 다양한 스프레드시트가 있을 수 있다. 각 스프레드시트는 "데이터 세트"로 불리어진다. 바람직한 실시예에서 사용되는 디바이스 데이터에 대한 스프레드시트는 아래에서 도 4a 내지 도 9를 결합하여 설명된다.

테스트 플로우는 또한 워크북내에 있는 스프레드시트로써 표현된다. 테스트 플로두는 실행되는 단계의 시리즈로써 특정된다. 각 단계에 관계된 데이터가 있고 각 단계는 스프레드시트에 있는 하나의 행으로써 표현된다. 각 단계에서 수행되는 정확한 동작은 하나이상의 플래그의 상태에 의해 영향을 받을 수 있다. 단계의 실행은 몇몇의 상기 플래그의 상태가 변경되게 할 수 있다. 이 경우에, 브랜칭 또는 다른 유사한 제어 구조가 이루어질 수 있다.

플로우에 있는 임의의 단계에 대해 특정된 데이터의 피스중의 하나는 어떤 테스트 템플릿이 어떤 데이터 세트와 사용될 것이라는 것이다. 테스트 템플릿과 상기 테스트 템플릿과 사용하기 위한 데이터 세트는 "인스턴스(instance)"라고 불리운다. 인스턴스는 또한 스프레드시트로써 설명될 수 있다. 바람직한 실시예에서 사용되는 플로우 정보에 대한 스프레드시트는 아래에서 도 10 내지 도 11을 결합하여 설명된다.

각각의 스프레드시트의 구조는 워크스테이션에 있는 소프트웨어에 의해 정의된다. 각 스프레드시트의 구조는 상업적 스프레드시트 프로그램의 공지된 메카니즘을 사용하여 템플릿을 특정함으로써 생성된다. 각 스프레드시트에 대한 데이터값은 테스트 엔지니어에 의해 채워지고 그 값은 테스트되는 디바이스의 타입에 의존할 것이다.

본 발명에 따른 테스터 제어 소프트웨어는 종래기술에 있는 것과 같이 디버그와 분석 툴 및 테스트 벡터 개발 툴을 사용할 것이다. 상기 용어들은 간소화를 위해 도 3에서는 도시되지 않는다. 바람직한 실시예에서, 벡터 파일은 너무 커서 쉽게 상업적인 스프레드시트 프로그램으로 조정할 수 없다. 이러한 이유로, Excel에서는 구현되지 않는다.

도 4a와 도 4b에서, 핀맵 디바이스 데이터 시트가 액세스되는 동안 워크스테이션(110)상의 디스플레이가 도시된다. 도 4a와 도 4b는 하나의 윈도우를 나타낸다. 윈도우는 워크스테이션(110)상의 Excel 소프트웨어에 의해 디스플레이된다. Excel 소프트웨어가 Windows NT(마이크로소프트사의 등록상표) 운영체제로 제어되기 때문에, 다중 윈도우가 한번에 스크린상에 디스플레이될 수 있다.

각 윈도우는 임의의 필드를 포함한다. 필드(410)는 툴바로써 알려져 있다. 필드(420)는 메뉴바로써 알려져 있다. 필드(430)는 데이터 셀 필드이다. 필드(440)는 탭 필드이다. 상기 필드의 컨텐트는 많은 경우에 워크스테이션(110)을 제어하기 위해 소프트웨어를 준비하는 부분으로써 특정된다. 그러나, 각각의 이러한 필드의 일반적인 동작 및 대체는 Excel을 사용하여 개발된 모든 애플리케이션에 대하여 표준이다. 따라서, 본 발명에서도, Excel이 반도체 테스트 시스템을 제어하기 위해 사용되고, Excel 또는 다른 Excel 애플리케이션에 친숙한 사람은 소프트웨어를 사용하기 위한 방법을 일반적으로 이해할 것이다. 따라서, 본 발명에서는 매우 쉽게 사용할 수 있기 때문에 커스텀 소프트웨어 시스템에 비교하여 상당한 시간 및 비용이 절약된다.

툴바(410)는 툴 아이콘(412,414)과 같은 많은 툴 아이콘을 포함한다. 아래에 상세히 설명되는 바와 같이, 테스터(112)용 테스트 프로그램은 다양한 스프레드시트에 데이터를 채움으로써 매우 특정화될 것이다. 데이터가 반도체 테스터에 대한 제어정보 또는 상업적 스프레드시트 프로그램이 정상적으로 사용되는 다른 타입의 데이터를 나타낸다는 사실에도 불구하고, 데이터는 동일한 방식으로 조정된다. 따라서, 데이터 조정 툴이 필요하지만, Excel로 사용되는 표준 데이터 조정 툴일 수 있다. 이러한 툴은 데이터를 카피하거나 삭제 또는 컴퓨터 파일에 스프레드시트를 저장하는 것과 같은 기능을 수행한다. 이러한 툴은 사용자가 특정 툴 아이콘을 선택하기 위해 마우스 또는 트랙볼과 같은 입력 디바이스를 조정함으로써 액세스된다.

부가하여, 필드(410)는 커스텀 툴로 커스터마이즈될 수 있다. Excel은 커스텀 툴이 툴바에 특정화되고 추가될 수 있게 한다. 원한다면, 테스트 프로그램의 생성에 유일한 기능을 수행하는 커스텀 툴이 툴바(410)에 추가될 수 있다. 예를 들면, 몇몇의 커스텀 툴은 다른 것들이 에러 또는 불일치에 대한 디바이스 데이터를 체크하는 동안 테스트 프로그램을 실행할 수 있다.

메뉴바(420)는 422와 424같은 시스템 메뉴 항목을 포함한다. 사용자에 의해 액세스될 때, 이러한 항목들은 시스템 메뉴 항목에 관련된 액션의 다중 선택을 제공하도록 확장된다. 예를 들면, FILE 시스템 메뉴 항목하에, 저장, 삭제 또는 프린트같은 선택이 있다. 툴바(410)와 메뉴바(420)로부터 액세스될 수 있는 툴들 사이에 어느정도의 오버랩이 존재한다. 일반적으로, 가장 평범한 툴은 툴바(410)상에 나타나고, 시스템 메뉴 항목들중의 한개하에서 선택중의 하나로써 종종 복사된다.

대부분의 메뉴 항목들은 자동 테스트 시스템에 사용되는 데이터에 특정하지 않은 기능을 나타낸다. 따라서, 디폴트 조건으로써 Excel 애플리케이션에 제공된 시스템 메뉴 항목은 대부분의 메뉴 항목으로 사용될 수 있다. 그러나, 자동 테스트 시스템에 특정한 많은 기능이 있다. 바람직한 실시예에서, 이러한 기능은 커스텀 메뉴 항목(426)으로부터의 선택을 사용자가 선택함으로써 실행된다. Excel은 상기 언급된 Excel 애플리케이션 개발 가이드에서 제공된 교시에 따라 애플리케이션에 대하여 커스터마이즈될 수 있다. 커스텀 메뉴 선택은 시스템 메뉴 항목으로써 동일한 메뉴바상에 또는 개별 커스텀 메뉴바상에 나타날 수 있다.

디바이스 데이터를 포함하는 데이터 시트에 대하여, 커스텀 메뉴 항목은 데이터 시트가 "타당성 검사"가 되도록 한다. 상기에 설명된 바와 같이, 데이터 시트에 있는 다양한 셀은 데이터 타당성 검사 기능과 관계될 수 있다. 이 데이터 타당성 검사 기능은 현재데이터 시트의 내용에 있는 부적절한 데이터 값을 검출할 수 있다. 예를 들면, 특정 필드가 수치값을 유지하도록 의도된다면, 입력값이 문자를 포함한다면 에러가 검출될 수 있다. 다른 셀이 양의 값을 제한되고 입력값이 음이라면 에러가 지시된다. 그러나, 몇몇의 타당성 검사 기능은 다중 시트에 의존한다. 예를 들면, 핀의 총수는 한개의 시트상에 특정화되고

임의의 속성은 다른 시트상의 각 핀에 대해 특정화될 수 있다. 속성이 너무 적은 핀에 특정화된다면, 이것은 두개의 시트를 단지 비교함으로써 검출될 수 있다. 다양한 타당성 검사 기능은 테스터 하드웨어의 제한과 요구에 근거하여 정의될 수 있다. 이 타당성 검사 기능은 커스텀 메뉴 항목으로부터 선택될 수 있다. 대안으로, 동일한 기능이 툴바 버튼을 사용하여 선택될 수 있다.

부가하여, 플로우 데이터 시트에 대하여, 플로우에 의하여 정의된 테스트를 실행하기 위한 방법이 있는 것이 요구된다. 플루오 시트와 관계된 커스텀 메뉴 항목은 플로우가 실행되고 또한 실행 방법의 원인이 된다. 예를 들면, 플로우의 실행에 중단점을 세팅하거나 또는 한번에 플로우 한 단계를 실행하는 것이 바람직하다.각각의 이러한 기능은 커스텀 메뉴 또는 툴바 버튼으로부터 메뉴 선택함으로써 특정화될 수 있다.

커스텀 메뉴 항목은 각각의 시트에 대하여 상이할 수 있다. Excel은 임의의 속성이 각 데이터 시트에 대하여 특정화되게 한다. 상기 속성중의 하나는 커스텀 메뉴 항목이다. 이러한 방식으로, 커스텀 메뉴는 특정 시트에 디스플레이된 특정 타입의 데이터에 적적한 동작을 포함하도록 만들어질 수 있다. 이후로, 테스터(112)를 제어하는 소프트웨어에 의해 수행되는 다른 동작이 설명된다. 이러한 동작은 커스텀 메뉴상에 나타난 메뉴 항목에 매핑되는 기능으로써 정의될 수 있다.

데이터 셀 필드(430)는 테스터(112)를 제어하기 위해 커스터마이즈된 데이터 셀을 포함한다. 셀 레이아웃은 Excel에 의해 제공된다. 그러나, 윈도우에서 볼 수 있는 행과 열의 수는 Excel을 프로그래밍함으로써 지시된다. 도 4a와 도 4b는 디바이스 핀맵 데이터 시트를 나타낸다. 따라서, 데이터 셀 필트(430)는 디바이스 핀맵 데이터를 수신하도록 레이아웃된 셀의 행과 열을 포함한다.

반도체 테스터에서, 디바이스 핀맵은 테스트 프로그램이 실행되는 동안 테스트 해야할 디바이스를 상기 핀(118)이 접촉하는 방법에 근거한 물리 핀(118;도1)에 논리 이름을 할당한다. 데이터 셀 필드(430)는 데이터 셀의 네개의 열(432a...432d)로 도시된다. 각각의 열은 헤더셀(434)과 같은 셀이 첫 머리에 주어진다. 헤더셀(434)은 헤더셀열의 잔여 셀에 채워질 데이터 타입의 설명을 포함한다. 스프레드시트가 Excel로 정의될 때, 임의의 셀은 헤더셀과 같이 고정값으로 주어질 수 있다.

데이터를 기입하고 체킹하는 규칙은 도 4a와 도 4b에 되시되지 않는다. 그러나, 상업적 스프레드 시트 프로그램의 한가지 특징은 다양한 셀에 기입된대로 타당성 검사를 할 수 있다는 것이다. 대안으로, 몇몇의 데이터 셀은 다른 셀에 기입된 값을 근거로 하여 계산될 수 있다. 막대한 양의 재무 데이터를 기입하기 위한 것과 같이 이러한 특징을 가진 상업적인 스프레드 시트가 개발된다. 그럼에도 불구하고, 데이터가 테스터를 제어하기 위해 사용될지라도, 몇몇의 데이터 기입 규칙이 가능하고 사용될 수 있다.

도 4a와 도 4b에 있는 또 다른 필드는 탭 필드(440)이다. 탭 필드(440)는 442와 444같은 탭의 시리즈를 포함한다. 각 탭은 테스트 프로그램을 포함하는 워크북내에 있는 하나의 스프레드시트를 나타낸다. 사용자는 스프레드시트가 탭중의 하나를 활동화함으로써 액티브상태에 있는 것을 선택한다. 도 4a와 도 4b의 도시에서, pinmap.pmp라는 이름을 가진 탭(443)은 도 4a와 도 4b에 도시된 핀 맵 스프레드 시트에 대응하는, 이것이 선택된 탭이다, 다른 탭의 상부위에 존재한다.

탭 필드(440)는 하나의 워크북으로써 모든 디바이스 데이터를 세팅하는 중요한 이점을 도시한다. 상이한 타입의 데이터가 상이한 툴로 기입되는 종래기술의 테스터와 대조적으로, 모든 디바이스 데이터는 한개의 Excel 워크북으로 기입된다. 한개의 워크북에 모든 데이터를 갖음으로써 한개의 스프레드의 데이터 값은 상이한 스프레드시트에 기입된 데이터에 의존하게 된다. 부가하여, 테스트 프로그램을 개발하는 사람은 상이한 타입의 데이터에 대한 시트간의 빠른 전환을 가능하게 한다.테스터에 대한 커스텀 프로그래밍 환경이 적절하게 구현된다면 각각의 상기 동작이 가능할 수 있는 반면, 이러한 단순성이 상대적으로 저비용과 하나의 프로그래밍 툴로부터 다른것으로 데이터를 통과하는 복잡성없이 이루어진다. 더우기, 이것은 프로그래머가 빨리 이해할 수 있도록 개념적으로 쉬운 형식으로 구현된다.

도 5에서는 제 2 디바이스 데이터 스프레드시트가 도시된다. 도 5는 채널 맵 데이터가 기입되는 동안 나타나는대로 워크스테이션(110)의 스크린상의 윈도우를 도시한다. 채널 맵 데이터가 핀 맵 데이터보다 상이함에도 불구하고 도 4a/4b와 도 5의 스크린 디스플레이는 매우 유사하다.

도 5의 스크린 디스플레이는 도 4a와 도 4b에서와 마찬가지의 필드를 갖는다. 툴 바(510)는 툴 바(410)과 동일하다. 메뉴 바(520)는 메뉴 바(420)과 동일하다. 메뉴 바(510)에 있는 몇몇의 메뉴 항목이 메뉴 바(410)에 있는 항목과 상이할 수 있는 것이 가능하다. 그러한 것은 만약 임의의 기능이 예를 들면 채널 맵 데이터에 단지 적용가능한 경우이다. 그 경우에, 상기 기능들에 대한 메뉴 항목 또는 서브 메뉴 항목은 메뉴 바(510)상에 단지 나타날 것이다. 바람직한 실시예에서, 디스플레이되는 스프레드시트에 있는 테스터 데이터의 타입에 의존하는 그러한 기능은 커스텀 메뉴 항목(526)하의 서브 메뉴 항목으로써 나타날 것이다. 대안으로, 데이터 의존 기능은 커스텀 툴 항목을 통하여 액세스될 수 있다.

테스터용 소프트웨어를 구현하도록 사용되는 상업적으로 이용가능한 스프레드시트는 상이한 타입의 데이터 시트가 정의되도록 하고 또한 디스플레이되는 서브 메뉴 항목이 사용된 데이터 시트의 타입에 의존하도록 하는 특징을 포함한다.

각 탭은 그 위에 두개의 부분 명칭을 갖는다는 것을 탭 필드(540)에서 주목하라. 제 1 부분은 스프레드시트에 대한 식별자이다. 제 2 부분의 명칭, 즉 확장자는 스프레드시트의 타입이다. 예를 들면, 도 5는 ChannelMap.cmp로 명명된 탭을 도시한다. 채널 맵을 정의하는 타입인 데이터를 유지하는 모든 스프레드시트는 확장명 ".cmp"를 가질 것이다. 그러나, 테스트 엔지니어는 특정 식별자를 선택한다. 바람직한 실시예에서, 워크스테이션(110)상에서 실행중인 상업적으로 이용가능한 스프레드시트 프로그램은 애플리케이션 개발자가 상기 타입의 스프레드시트에 대해 메뉴 바(420)상에 메뉴 항목과 서브 메뉴 항목이 나타나는 것을 특정화하게 한다.

도 6에서, 스펙 시트에 대한 스프레드시트가 도시된다. 모든 다른 데이터 툴을 가짐에 따라, 상기 스프레드시트는 툴 바(610), 메뉴 바(620) 및 탭 필드(640)를 갖는다. 탭 필드(640)는 "SpecSheet.sps"로 명명되어 강조된 현재의 스프레드시트와 함께 도시된다. 게다가, 스펙시트는 행과 열로 구성된 많은 데이

터 셀을 갖는 데이터 필드(630)를 갖는다. 여기에서, 각 행은 특정한 테스트중인 디바이스와 관계된 파라미터에 대한 심볼을 포함한다. 다양한 정보가 각 파라미터에 대하여 주어진다. 도 6은 데이터 필드(630)의 각 행에 7개의 열을 보여준다. 심볼 명칭에 부가하여, 각 행은 값, 그 값에 대한 단위, 심볼의 설명에 대한 텍스트 필드, 적절하게 동작하는 최소 및 최대값, 및 코멘트 필드를 포함한다. 데이터 시트가 단일 Excel 워크북에서 모두 개발되기 때문에, 다른 스프레드시트에서 심볼이라는 이름으로 언급될 수 있다. 더 중요하게도, 각 심볼에 관계된 파라미터 또한 다른 스프레드시트에서 사용될 수 있다. 부가적으로, 스펙시트라는 명칭은 테스트 인스턴스가 개발됨에 따라 특정 테스트 템플릿으로 파라미터로써 통과될 수 있다. 이것은 테스트 템플릿이 테스트되는 디바이스가 명세(즉, 테스트 통과 또는 실패)내에서 동작하는지를 결정하기 위한 것과 같이 스펙시트 데이터에 비교하게 한다.

도 6은 단일 명세 시트를 도시한다. 단일 작업내에서 상이한 타입의 테스트에대하여 상이한 명세가 요구되는 것이 가능하다. 따라서, 다중 명세 시트가 사용될 수 있다. 데이터 관리자(316)과 결합하여 아래에 설명되는 바와 같이, 특정 타입 데이터의 다중 세트를 갖는 규정이 만들어진다.

도 7은 에지 세트를 정의하기 위해 사용되는 추가의 디바이스 데이터 시트를 도시한다. 도 1에 도시되는 바와 같이, 테스터(112)는 다중 핀(124)을 포함한다. 각 핀은, 각각 에지를 발생시키며 에지 제너레이터로 불리는다중 회로를 포함한다. 에지는 핀(124)에 의한 어떤 행동에 시간을 부여하기 위해 사용되는 신호이다. 예를 들면, 핀에 의해 발생된 펄스의 시작과 중지 시간에 따라 그러한 파라미터를 제어하는 4개의 에지 신호가 존재할 수 있다. 각 에지가 발생된느 시간이 프로그래밍될 수 있다. 하나의 핀에 대한 각 에지가 프로그래밍되는 시간은 에지 세트로써 정의된다. 에지가 특정 핀에 대하여 프로그래밍되는 시간이 작업동안 변할 수 있기 때문에, 각 핀에 대하여 다중 에지가 존재할 수 있다.

도 7은 하나의 핀에 대한 에지 세트를 도시한다. 열(750)은 에지 세트가 특정되는 핀이 "QDataPins"로 불린다는 것을 보여준다. 다른 핀들에 대한 에지 세트는 더 많은 행을 데이터 필드(730)에 부가함으로써 부가될 수 있다. 열(752)은 핀 QDataPins에 대한 하나의 에지 세트가 "LooseFunctions1"으로 불린다는 것을 보여준다. 다른 에지 세트가 더 많은 행을 데이터 필드(730)로 부가함으로써 정의될 수 있다.

열(754,756)은 원하는 신호를 발생시키기 위해 핀(124)을 셋업하는데 필요한 다른 정보를 제공한다. 열(758)은 핀 및 인접한 열(760,762)내에 있는 에지를 목록화하고 에지가 발생하는 시간에 대하여 값을 공급한다.

열(760,762)는 상업적으로 이용가능한 스프레드시트를 구비한 테스터(112)를 제어하는데 유용한 이점을 보여준다. 스프레드시트는 등식으로부터 값을 자동으로 계산할 수 있다. 따라서, 열(762)은 각 에지에 대한 시간 값을 다른 스프레드시트에서 정의된 파라미터 또는 다른 상수와 관련한 등식으로써 특정되게 한다. 열(760)은 필드가 열(762)에 있는 등식을 평가함으로서 프로그래밍된 스프레드시트 프로그램에 의해 자동으로 계산되는 수치값을 도시한다.

셀(766,768,770 및 772)은 상업적으로 이용가능한 스프레드시트 프로그램을 사용함으로써 얻어질 수 있는 다른 이점을 보여준다. 데이터 필드(730)에 있는 드롭 다운 리스트 상자는 데이터 필드(730)에 있는 데이터 셀의 값 또는 디스플레이를 제어하기 위해 사용될 수 있다. 드롭 다운 리스트 상자(766,768,770 및 772)가 이러한 형식으로 사용될 수 있다. 상기 셀은 드롭 다운 상자로써 구현되고, Excel에 있는 표준 객체이다. 각 드롭 다운 상자는 고정 수치값중의 하나를 가질 수 있다. 상기 값은 워크북내에 있는 스프레드시트중의 하나에 기입된 값에 기준으로써 또는 고정 목록으로써 특정화될 수 있고, 다음에 기입된 값이 변함에 따라 변할 수 있다.

테스터(112)에 많은 수의 핀이 주어지면, 에지 세트 스프레드시는 잠재적으로 많은 행을 포함할 수 있다. 셀(766,768,770 및 772)은 한번에 디스플레이되는 데이터 양을 제한하기 위해 사용될 수 있다. 셀(766)은 특정 핀 또는 핀 그룹의 이름을 포함할 수 있다. 핀 그룹내에 있는 핀 또는 상기 핀에 대해 단지 에지 세트는 데이터 필드(730)에 디스플레이된다. 유사한 방법으로, 셀(768,770)은 특정 타이밍 모드에 대한 에지 세트와 에지의 단지 선택된 수에 대한 에지 세트를 디스플레이한다. 셀(772)은 유사하게 드롭 다운 상자이지만, 값 열(760)에 기입된 값에 대한 단위를 특정화하기 위해 사용된다.

도 8a는 Excel 워크북내에 있는 또 다른 데이터 시트를 도시한다. 도 8a에 있는 시트는 타임 세트에 대한 데이터를 제공하고 다른 데이터 시트처럼 동일한 포맷으로 존재한다. 타임 세트는 핀(124)의 각각에 대한 하나의 에지 세트의 집합이다. 다중 타임 세트가 종종 정의되어 상이한 프로그래밍된 타임이 작업내에 사용될 수 있다.

도 8a는 타임 세트와 핀 또는 핀 그룹과 관계된 주기를 타임 세트라는 이름으로 데이터의 엔트리를 할당하는 열을 도시한다. 각 핀 또는 핀 그룹에 대하여, 테이블에 하나의 행이 있다. 각 핀 또는 핀 그룹에 대하여, 핀의 에지에 대한 프로그래밍된 타임이 에지 세트에 대한 참조로써 또는 등식으로 특정화되게 하는 열이 있다. 도 8a는 또한 드롭 다운 상자(832a,834a)를 도시한다. 드롭 다운 상자(832a)는 어떤 타임 세트가 디스플레이될지를 선택한다. 드롭 다운 상자(834a)는 모든 타임 값이 디스플레이되는 단위를 특정화한다.

도 8b는 상업적으로 이용가능한 스프레드시트 프로그램을 사용하여 발생하는 또 다른 이점을 도시한다. 다중 스프레드시트 템플릿이 동일한 데이터에 대하여 정의될 수 있다. 동일한 스프레드시트에 대해 다중 템플릿을 갖는 것은 사용자가 작업중인 어떤 작업일지라도 가장 유용한 포맷으로 사용자가 볼 수 있게 하기 위하여 동일한 데이터를 상이하게 디스플레이되도록 한다. 도 8b는 에지 세트 데이터를 디스플레이하는 대체 템플릿을 도시한다. 드롭 다운 상자(832a)와 같은 드롭 다운 상자(832b)가 존재한다. 또한 데이터 필드(830a)와 유사하게 데이터 필드(830b)를 갖는다. 그러나, 데이터 필드(830a,830b)에 있는 열은 상이하게 구성된다.

도 8a에서 도출된 템플릿은 예를 들면 핀 그룹을 특정 타임 세트로 할당하기 위해 사용될 수 있다. 도 8b에 있는 템플릿은 예를 들면 각 핀이 어떤 에지 세트로 할당되는지를 결정하기 위해 사용될 수 있다. 바람직한 실시예에서, 도 8b에 있는 템플릿은 판독 전용 템플릿일 수 있다.

도 9는 확정된 포맷에 사용자가 쉽게 기입할 수 있는 또 다른 타입의 데이터를 도시한다. 도 9는 다른 스프레드시트에서 사용되는 것과 같이 동일한 툴 바와 메뉴 바를 도시한다는 것을 주목하라. 도 9는 다양하게 명명된 파라미터와 관계된 레벨을 특정화하는 스프레드시트를 도시한다. 파라미터는 다양한 측정 또는 자극 신호와 관계된 알맞은 전압 및 전류 레벨이다. Excel은 상기 명명된 파라미터가 다른 스프레드시트에 있는 이름에 의해 참조될 수 있게한다. 또한 파라미터의 값이 다른 명명된 값을 사용하는 등식에 의해 특정화될 수 있게 한다.

도 10a에서, 본 발명의 이점중의 하나를 볼 수 있다. 도 10a는 "플로우 타입" 스프레드시트를 나타낸다. 도 10a는 테스트 "인스턴스"가 특정 테스트에 대하여 정의되는 방법을 나타낸다. 반도체 테스트 시스템은 일반적으로 제한된 수의 물건에 대하여 테스트하는 것을 필요로 한다. 예를 들면, 테스트 시스템은 패턴 "버스트"를 실행할 수 있다. 버스트 동안, 테스터는 특정 자극을 인가하고 임의의 기대 응답에 대하여 체크한다. 자극과 응답에 대한 값은 "벡터 파일"로 특정된다. 벡터 파일에 있는 상이한 값은 상이한 종류의 부분을 테스트하거나 또는 동일한 부분상의 상이한 테스트를 실행하기 위하여 사용된다. 그러나, 테스터가 벡터 파일 사용에 대해 검사하는 동작은 그 안의 특정 데이터 값에 관계없이 일반적으로 동일하다.

테스트 시스템이 패턴 버스트를 실행하는 것을 검사해야 하는 연속의 단계는 "템플릿"으로써 코딩된다. 템플릿은 특정 벡터 파일의 설명에 대한 블랭크 스페이스 또는 필요한 다른 데이터와 동등한 프로그램으로 실행되어야 하는 일련의 단계이다. 이러한 방식으로, 템플릿은 일단 사용될 특정 데이터 값이 특정화되면 임의의 테스트에 대해 사용될 수 있다. 일단 데이터가 특정 템플릿을 사용하기 위하여 특정되었다면, 템플릿의 "인스턴스"가 발생된다. 인스턴스는 프로그램에서 사용될 수 있다.

테스터 제조업자는 일반적으로 제어 소프트웨어의 부분으로써 몇개의 테스트 템플릿을 제공한다. 테스트 템플릿은 패턴 버스트를 실행하거나 또는 누설 전류를 측정하하기 위해 사용될 수 있고, 테스트 해야할 디바이스의 리드상의 DC 레벨 또는 다른 테스터 동작을 검사하기 위해 사용될 수 있다. 아래에 더 상세하게 설명되는 바와 같이, 바람직한 실시예의 템플릿은 비쥬얼 베이직으로 기록되며, 비쥬얼 베이직은 Excel 스프레드시트 프로그램과 관계된 프로그래밍 언어이다. 각각의 템플릿은 비쥬얼 베이직 과정으로써 기록되며 데이터 값은 그 과정에 대한 인수(arguments)로써 특정화된다.

도 10a는 프로그램 인스턴스가 발생되는 방법을 도시한다. 데이터 필드(1030)는 테스트 작업에서 사용될 수 있는 인스턴스의 명칭을 목록화하는 열(1032a)을 포함한다. 열(1032b)은 인스턴스를 발생하기 위해 사용되는 템플릿의 명칭을 포함한다.

도 10b는 특정 데이터가 얼마나 템플릿과 관계가 있는지를 보여준다. Excel은 대화상자(1036)가 데이터 필드(1030)내에 있느 특정 셀과 관계가 있도록 하는 프로그래밍 특징을 포함한다. 상이한 대화상자가 셀 또는 몇몇의 다른 셀에 있는 값에 의존하면서 선택될 수 있다. 예를 들면, 도 10b에서, 대화상자(1036)이 셀(1034)와 관련하여 도시된다. 셀(1034)은 특정 데이터 값을 셀(1035)에 목록화된 템플릿에 관계시킴으로써 발생되도록 하는 인스턴스의 명칭이다. 상기 예에서, 대화상자(1036)의 포맷은 셀(1035)에 목록화된 템플릿에 근거하여 선택된다.

대화상자(1036)는 영역(1038,1050)을 포함하며, 임의의 데이터 값이 셀(1035)에 있는 템플릿과 관계되는 필드를 구비한 각각이 목록화된다. 영역(1038)은 요구 필드를 포함한다. 영역(1050)은 옵션 필드를 포함한다. "30Mhzfunc"로 명명된 테스트 인스턴스를 발생시키기 위해, 대화상자(1036)에 있는 값은 "functional"로 명명된 템플릿에 대한 인수로써 통과된다.

도 10b는 데이터 필드(1030)가 많은 행을 포함하고 있다는 것을 도시한다. 테스트 작업에 필요한 각 테스트 인스턴스에 대해 하나의 행이 존재한다. "functional"로 명명된 템플릿이 두개의 상이한 인스턴스를 발생시키기 위해 사용된다는 것이 주목된다. 상이한 인스턴스는 대화상자(1036)에 있는 상이한 인수를 특정화함으로써 동일한 템플릿을 가지고 발생된다.

도 10a와 도 10b는 테스트 플로우를 특정화하는 단계를 도시하고 있다. 그러나, 사용자 인터페이스는 도 4 내지 도 9의 데이터 스프레드시트에 대한 사용자 인터페이스와 동일하다는 것이 주목된다. 따라서, 프로그램 플로우를 특정화하기 위해 특정 트레이닝이 요구되지 않는다. 또한 대화상자(1036)에 기록된 인수는 다른 스프레드시트의 명칭 또는 스프레드시트내에 정의된 값이다. 따라서, 디바이스 데이터는 소프트웨어 개발 시간 절약을 고려하여 프로그램 플로우 정보와 쉽게 결합된다. 이러한 이점은 디바이스 데이터와 데이터 프로그램이 스프레드시트로 표현된다는 사실에서 얻어진다.

다른 프로그램 플로우 정보는 도 11a에 나타난 스프레드시트에 기입된다. 일단 인스턴스가 정의되면, 인스턴스가 실행되는 순서가 특정화되어야 한다. 부가하여, 인스턴스가 테스트 작업내의 테스트를 설명하기 때문에, 테스트 인스턴스가 통과 또는 실패인지 발생하는 이벤트에 정보가 또한 제공되어야 한다. 이러한 모든 정보는 다른 플로우와 디바이스 데이터 스프레드시트와 같이 동일한 인터페이스를 사용하는 스프레드시트에 사용자에 의해 기입된다.

데이터 필드(1130)내에 테스트 작업에서 각각의 단계에 대해 특정되어야 하는 정보를 유지하는 몇개의 열이 존재한다. 데이터 필드(1130)에 있는 각 행은 테스트 작업에서 하나의 단계상에 정보를 유지한다. 열(132)는 "OPCODE"로 명명된다. 이 열은 특정 단계에서 발생하는 것을 가리킨다. 일반적으로, OPCODE 열(132)에 있는 각 엔트리는 테스트 인스턴스이다.

열(1134)은 테스트에서의 단계 수를 포함한다. 단계 수는 테스트의 실행을 보고하기 위해 테스트 시스템에 의해 사용될 수 있다. 예를 들면, 결점이 테스트동안 검출된다면, 검출이 검출된 단계수가 보고될 수 있다.

열(1136)은 통과 빈 수를 포함한다. 각 디바이스가 테스트됨에 따라, 테스터는 일반적으로 테스트의 결과에 대응하는 "빈(bin)" 수로 할당한다. 빈 수는 부분의 그레이딩을 가리킨다. 예를 들면, 몇몇의 빈 수는 디바이스 명세로 동작하는 부분을 나타낸다. 다른 빈 수는 결점이 있어서 소용이 없는 부분을 가리킬 수 있다. 그러나, 다른 빈 수는 명세내에서 수행하는 것이 아니라 감소된 명세로 동작하기에 충분한

기능으로 동작하는 디바이스를 가리킬 수 있다. 테스트 엔지니어는 빈 수와 그것의 중요도를 할당한다.

열(1138)은 유사하다. 테스트가 실패한다면 할당된 빈의 수를 포함한다.

데이터 필드(1130)는 도시된 것 이상의 다른 열을 포함할 수 있다. 도 11b는 더 복잡한 플로우 제어를 위해 정보의 더 많은 열로 확장된 데이터 필드(1130b)를 갖는 플로우 시트를 도시한다. 예를 들면, 데이터 필드(1130b)는 플래그 필드(1156,1158)를 포함한다. 플래그 필드(1156)는 임의의 단계에서 특정된 테스트가 통과한다면 "참"으로 세팅되는 동일한 플래그, 즉 불린(boolean) 변수를 선택적으로 포함한다. 필드(1158)는 유사하지만, 테스트가 실패한다면 세팅되는 플래그의 이름을 포함한다.

데이터 필드(1130b)에 있는 다른 열은 심지어 다중 디바이스가 동시에 테스트되는 경우의 더 복잡한 플로를 위해 사용될 수 있다. 예를 들면, 열(1152)는 임의의 행에 특정된 단계에서 적용될 수 있는 디바이스의 단계를 가리킨다. 특정 테스트는 모든 디바이스에 적용될 수 있다. 다른 테스트가 모든 이전 테스트를 통과한 디바이스에 단지 실행될 수 있다. 다른 테스트는 이전 테스트를 실패한 디바이스상에 단지 실행될 수 있다.

열(1154)은 테스트의 결과가 적용되는 디바이스를 특정화한다.

도시되지 않은 다른 열이 또한 사용될 수 있다. 상기 단계에서 테스트 인스턴스의 실행에 앞서 불린 표현으로 평가되는 조건 필드를 포함하는 열이 존재할 수 있다. 이러한 방식으로, 조건적 프로그램 플로우가 구현될 수 있다. 이러한 기능은 부울식을 수용하게 함으로써 열(1152)과 조합될 수 있다. 참(true)으로 산출하는 부울식은 모든 디바이스에서 시험이 행해질 수 있다는 것을 표시한다. 역으로, 부울식이 거짓으로 산출되었다면, 어떠한 디바이스에서도 시험이 행해지지 않을 것이다. 부울식이 각각의 디바이스에 대하여 명시될 수 있다면, 하이브리드 위치가 또한 가능한다. 도 11b의 예에서, 열(1152)은 상수 "all"을 가진 몇몇 값을 포함한다. 이러한 값은 행이 기준인 시험이 무조건으로 모든 디바이스에서 행해져야 한다는 것을 표시한다. 그러나, 다른 행은 앞선 행의 열(1154 또는 1158)에 리스트된 플래그를 기준으로 하는 부울식을 도시한다. 추가로, 디버그 정지 지점처럼 작용하는 특정값으로 셀이 설정될 수 있는 열을 포함할 수 있다.

도 11 b에 도시된 정보는 현존 테스터를 프로그램하도록 명시되어야 하는 정보와 유사하다. 그러나, 도 11b의 플로우 정보는, 각각의 행이 각각의 열내에 여러 정보를 가진 열을 포함하고 있는 상태에서, 행으로 조직화되어 있다. 플로우 정보가 텍스트 파일에 저장된 것과 비교하여, 상업적으로 이용가능한 스프레드 시트 프로그램을 이용한 행과 열의 조직은 프로그래머에게는 보다 직관적이기에, 프로그래머는 보다 빨리 플로우를 프로그래밍하는 것을 배울 수 있다. 제 2 장점은 도 11b에 도시된 플로우 정보를 조작하는 구조와 툴은 상업적으로 이용가능한 스프레드 시트 프로그램에 의해 제공된다는 것이다. 따라서, 소프트웨어가 신속하게 개발될 수 있다.

도 11c는 플로우 제어 정보가 명시될 수 있는 대체 방법을 도시하고 있다. 도 11c는 낮은 제어 소프트웨어(314)(도 3)의 옵션을 수정하는 옵션을 명시하는데 사용될 수 있는 팝업 옵션 박스를 도시하고 있다. 대화 박스내의 선택부를 클릭함으로써 옵션은 시험 엔지니어에 의해 명시된다. 예를 들어, "Do All" 옵션으로, 플로우 제어 소프트웨어(314)는 일단계가 실패하더라도 프로그램 플로우내의 모든 단계를 실행할 수 있다. 이러한 옵션이 선택되지 않은 경우에, 하나의 시험이 실패하는 즉시 특정 부분의 시험이 중지한다. 다른 예로서, "Print Final" 옵션으로, 시험 작업이 완료되는 즉시 시험의 결과가 인쇄된다. 선택되지 않은 경우에, 그 결과는 사용자가 판독할 때까지 인쇄하지 않고 저장하고 있다. 또 다른 예로서, "Do Time" 옵션으로, 플로우 제어 소프트웨어는 작업의 시작 시간과 작업의 종료 시간을 기록할 수 있어서, 작업에 걸리는 시간의 길이를 기록할 수 있다. 아래에 설명된 바와 같이, 플로우 데이터 시트는 플로우 데이터 구조로 변환된다. 명시된 옵션은 변환이 행해지는 정확한 방법에 영향을 준다.

도 12a와 도 12b는 시험 템플릿의 샘플을 도시하고 있다. 시험 템플릿은 스프레드 시트에 부착된 매크로를 기록하는데 사용될 수 있는 프로그래밍 언어, 즉 비쥬얼 베이직으로 기록된다. 각각의 매크로는 하나의 절차로서 프로그래밍 언어로 사용될 수 있고, 시험 템플릿으로서 사용될 수 있다.

도 12a와 도 12b에서 일부 중요한 점을 발견할 수 있다. 먼저, 전체적인 프로그래밍 환경은 디바이스 데이터와 플로우 스프레드 시트에 정보를 입력하는데 사용된 것과 동일하다. 동일 툴 바(1210), 메뉴 바(1220), 및 탭 필드(1240)를 가지고 있다. 데이터 필드대신에, 시험 템플릿 스크린은 프로그램 코드 라인을 포함한다.

프로그램 코드의 각 라인은 테스터 제조업자에 의해 개발된 다른 절차를 호출할 수 있다. 라인(1250,1254)은 테스터 제조업자에 의해 제공된 기능에서 호출을 나타낸다. 라인(1250)에서, "PinHi"는 디바이스 드라이버(328)(도 3)중 하나에 액세스하는 절차이고, 명시된 핀을 하이 상태로 설정한다. 라인(1254)에서, "StartPatgen"은 타이밍 및 시퀀스 회로(120)(도 1)를 동작시키는 유사한 기능이다.

도 12a와 도 12b는 또한 디버그 툴 바(1232)를 도시한다. 새로운 반도체 디바이스가 먼저 개발되어 있을 때, 디바이스가 적당히 설계되었는지를 평가하기 위해 반도체 디바이스 엔지니어가 디바이스의 성능을 측정하는 것이 필요하다. 결함이 검출되는 경우에, 디바이스 엔지니어는 무엇이 잘못되었는지를 이해하고자 한다. 이렇게 하기 위해서는, 디바이스 엔지니어는 "디버그 모드"의 테스터를 이용한다. 디바이스 엔지니어는 하나의 동작을 한번에 실행하기 위해 테스터를 제어함으로써, 각각의 동작의 결과가 분석될 수 있다. 디버그 툴 바(1232)는 디버그 동작동안에 사용되는 툴을 포함한다. 디버그 툴 바(1232)는 상업적으로 이용가능한 스트레드 시트 프로그램의 일부로서 바람직하게 제공된다.

디버그 툴 바(1232)는 여러 디버그 기능을 수행하는 툴을 포함한다. 예를 들어, 툴(1234)은 코드내의 정지 지점을 설정한다. 라인(1254)은 정지 지점이 그 라인에 설정됨을 표시하기 위해 하이라이트된다. 시험동안에, 정지 지점으로 마킹된 라인의 실행전에, 실행이 중지된다. 그러면, 디바이스 엔지니어는 테스터 또는 디바이스의 상태를 체크할 수 있다.

정지 지점에서 실행이 중지되는 즉시, 디버그 툴 바(1232)상의 다른 툴은 실행이 계속되는 방법을 제어한다. 예를 들어, 툴(1236)로, 추후의 모든 절차로 점진적으로 진행하도록 실행할 수 있다. 대조적으로,

다른 툴로, 절차가 건너 뛸 수 있고, 또 다른 툴로 특정 값이 조사될 수 있다. 이러한 방식으로, 디바이스 엔지니어는 디바이스의 동작에 대하여 상당히 관찰할 수 있다. 동일 디버그 툴이 또한 피시험 디바이스용 시험을 개발하는 테스터 엔지니어에 의해 사용될 수 있다. 상당히, 이러한 기능의 대부분은 디버그 툴로 프로그래밍 언어와 관련된 상업적인 스트레드 시트 프로그램을 이용함으로써 상당히 저가로 제공된다.

시험작업용 데이터와 플로우 정보가 시험 엔지니어 또는 다른 시스템 사용자에 의해 입력되는 즉시, 워크 스테이션(110)의 소프트웨어는 시험 작업을 제어하도록 동작될 수 있다. 도 3에 도시된 바와 같이, 사용자의 입력에 응답하여 시험 작업은 플로우 제어 소프트웨어(314)에 의해 실행된다. 예시된 실시예에서, 플로우 제어 소프트웨어(314)는 C++ 프로그램으로서 구현된다. 보다 상세하게는 DLL이다.

도 13은 플로우 제어 소프트웨어(314)의 동작을 설명하는 플로우 챠트이다. 플로우 제어 소프트웨어(314)는 아이들 상태(1310)로 시작한다. 플로우 제어 소프트웨어(314)는 그 소프트웨어로의 입력에 따라 3개의 상이한 동작중 하나를 수행할 수 있다. 플로우 제어 소프트웨어(314)가 행하는 하나의 동작은 시험 작업동안에 액세스될 수 있는 데이터 구조로 시험 작업용 데이터를 조직화하는 것이다. 도 10과 도 11와 결부시켜 설명된 바와 같이, 시험 작업은 인스턴스의 리스트로서 명시되어 있고, 인스턴스는 특정 디바이스의 특정 템플릿과의 연관성에 의해 명시되어 있다. 추가로, 특정 인스턴스가 실행되는 순서는 프로그램 플로우 데이터 시트로 입력된 조건에 영향을 받을 수 있다.

사용자의 명령에 응답하여, 플로우 제어 소프트웨어(314)는 실행되어야 하는 각각의 템플릿과 그 템플릿의 실행에 사용되어야 하는 데이터를 리스트하는 작업 제어 데이터 구조를 만든다. 템플릿의 실행 순서가 조건으로부터 결정될 수 있는 곳에서, 템플릿이 작업 제어 데이터 구조로 배치되기 전에 결정된다. 그러나, 템플릿의 실행 순서가 시험 작업이 실행되기 전까지 결정되지 않은 곳에서, 작업 제어 데이터 구조는 템플릿이 실행되는 순서를 결정하도록 값이 정해진 조건의 정보를 단순히 포함한다. 작업 제어 데이터 구조는 색인 리스트일 수 있지만, 다른 것이 가능하다.

작업 제어 데이터 구조를 만들 때, 단계(1312)로 실행한다. 단계(1312)에서, 제 1 템플릿이 결정된다. 그 템플릿은 플로우 데이터 시트내의 제 1 인스턴스로부터 결정된다. 이러한 인스턴스와 연관된 템플릿은 인스턴스 데이터 시트로부터 결정된다. 그 템플릿용 기호가 작업 제어 데이터 구조에 입력된다. 바람직한 실시예에서, 기호는 템플릿을 실행하도록 호출하는 서브 프로그램의 이름를 나타낸다.

단계(1314)에서, 선택된 템플릿 서브프로그램의 호출시 패스되는 데이터가 확인된다. 어떤 데이터가 사용되는지는 인스턴스 데이터 시트로부터 결정된다.

단계(1316)에서, 명시된 시험 템플릿용으로 사용되어야 하는 데이터의 확인이 작업 제어 데이터 구조에 추가된다. 도 3에 도시된 바와 같이, 데이터 매니저는 테스터(112)(도 1)에 적용하기 위해 Excel 워크북에서 디바이스 드라이버로 데이터를 패스한다. 아래에 상세히 설명되는 바와 같이, 데이터 매니저(316)는 시험 작업동안에 테스터(112)로 패스되는 특정 데이터를 수용하는 파라미터 데이터 구조를 짜맞춘다. 파라미터 데이터 구조내의 데이터는 "콘테이너"로 조직화되어 있어서, 올바른 "콘테이너"의 이름을 데이터에 제공함으로써 파라미터 데이터 구조로부터 올바른 데이터가 선택될 수 있다. 플로우 제어 데이터 구조에 부가되어야 하는 인수는 데이터 매니저(316)가 파라미터 데이터 구조로부터 선택하는 올바른 "콘테이너"를 명시한다. 동일 텍스트 스트링은 콘테이너와 관련되어 있다. 그러나, 콘테이너를 확인하는 다른 방법이 선택적으로 사용될 수 있다.

"콘테이너"의 이름을 나타내는 인수에 추가로, 데이터 값인 인수는 특정 템플릿에 의해 또한 사용될 수 있고, 파라미터 데이터 구조에 또한 부가될 수 있다.

템플릿과 데이터 값을 확인하는 정보가 작업 제어 데이터 구조에 부가되면, 단계(1318)로 실행된다. 단계(1318)에서, 플로우 시트에 명시된 다음 템플릿을 실행하는 조건이 결정된다. 그 조건은 실행할 다음 인스턴스가 될 수 있는 플로우 시트내의 모든 가능한 인스턴스를 결정하도록 연산된다. 추가로, 각각의 인스턴스가 다음일 수 있는 조건은 논리 방정식으로 표현된다. 변수는 시험 작업동안에 결정되어야 하는 값으로 사용된다. 이러한 논리식은 단계(1320)에서 작업 제어 데이터 구조로 입력된다. 논리식은 무언가가 종래의 프로그래밍 언어내의 브렌치 명령과 동일하게 할 수 있는 조건으로서 사용된다.

플로우 제어 시트의 하나의 행에 대한 정보가 입력되면, 실행은 단계(1322)로 진행한다. 단계(1322)에서, 플로우 제어 데이터 시트내의 추가 열이 있는지를 체크한다. 있다면, 실행은 단계(1312)로 진행한다. 플로우 제어 데이터 시트내에 추가 행이 없다면, 플로우 제어 소프트웨어(314)는 아이들 상태(1310)로 복원한다.

플로우 제어 소프트웨어(314)가 아이들 상태로 있는 경우에, 시험 작업을 행하기 위해 사용자로부터 명령을 수신하거나, 피시험 디바이스가 로딩되거나 시험준비에 있다는 것을 표시하는 핸들링 디바이스(114)로부터의 시작 신호를 수신한다. 시험 작업을 행하는 명령이 수신될 때, 실행은 단계(1324)로 진행한다. 단계 (1324)에서, 작업 제어 데이터 구조로부터의 제 1 템플릿이 판독된다. 그 템플릿은 작업 제어 데이터 구조로부터의 명시된 인수를 가진 서브프로그램으로 불리운다. 그 템플릿 서브프로그램은 실행하고 플로우 제어 데이터 구조(314)로 복원한다.

시험을 나타내는 각각의 템플릿 서브프로그램은 시험이 패스 또는 실패한지를 표시한다. 시험 결과가 표시될 수 있는 한가지 방법은 그 템플릿 서브프로그램에 의해 행해지는 시험 결과를 표시하는 호출 프로그램에서 하나의 값을 복원하도록 템플릿 서브프로그램이 코드화되는가이다. 대안으로, 템플릿 또는 적절하다면 디바이스 드라이버는 시험이 패스되거나 실패되었는지를 결정하도록 테스터의 상태를 체크하는 플로우 DLL내의 기능을 호출할 수 있다. 단계(1326)에서, 시험 작업의 상태는 바로 호출된 시험 템플릿에 의해 복원되는 값을 근거로 갱신된다. 예를 들어, 시험은 시험 템플릿이 에러없이 실행되었는지(즉, 시험이 패스) 또는 에러가 검출되었는지를 표시할 수 있다. 일부 인스턴스에서, 에러는 피시험 디바이스의 격납을 용이하게 하는 수를 가질 것이다.

단계(1328)에서, 단계(1328)에서 작업 플로우 데이터 구조에 저장된 논리식을 결정하는데 시험의 새로운

상태가 사용된다. 그 결정을 근거로 하여, 다음 템플릿이 단계(1320)에서 저장된 브렌치 정보를 근거로 하여 선택된다.

그 다음 실행은 단계(1330)로 진행한다. 단계(1330)는 실행할 추가 템플릿이 존재하는가를 결정한다. 있다면, 작업 실행은 단계(1324)로 역으로 진행한다. 추가 템플릿이 없다면, 플로우 제어 소프트웨어(314)는 아이들 상태로 보원한다.

바람직한 실시예에서, 시험 작업의 결과는 시험이 완료할 때 사용자에게 또는 핸드링 디바이스에 제공되지 않는다. 데이터가 주어지면, 시험 작업의 실행이 느려진다. 그러므로, 바람직한 실시예에서, 각 시험 작업의 실행 결과는 요구가 행해질 때까지 컴퓨터 메모리에 저장된다. 결과에 대한 요구로, 플로우 제어 소프트웨어(314)는 아이들 상태(1310)에서 단계(1332)로 패스할 수 있다. 단계(1332)에서, 그 결과는 사용자를 위해 화면표시될 수 있거나, 핸들링 디바이스(114)로 패스된다. 그 결과는 사용자에게 편리한 또는 핸들링 디바이스에 적절한 방식으로 화면표시를 위해 포맷될 수 있다.

데이터 매니저(316)에 대하여 보다 상세히 설명될 것이다. 데이터 매니저 (316)는 일련의 C++ "클래스"로 구성되어 있다. 클래스는 관련된 상태도를 가지고 있지 않다. 클래스는 다른 프로그래밍 구성 요소로부터의 요구에 응답하여 동작한다. 클래스는 관련된 "방법"을 가지고 있다. 각각의 방법은 클래스내에 저장된 데이터에 액세스하는 종래의 프로그래밍 기능과 같이 호출될 수 있다.

C++ 방법은 다른 C++ 프로그램에 의해 바로 호출될 수 있다. 도 3의 바람직한 실시예에서, 디바이스 드라이버(328)는 C++ 절차에 의해 구현될 수 있기 때문에, 데이터 매니저(316)내의 데이터를 바로 판독하거나 변경하도록 그 방법을 호출할 수 있다. C++ 가 아닌 액셀 워크북은 데이터 매니저(316)으로부터 데이터를 저장하거나 검색하는 방법을 호출하기 위해 API-애플리케이션 프로그램 인터페이스를 필요로 한다. 그러나, API는 공지된 프로그래밍 구조이다. 비쥬얼 베이직을 포함한 액셀 프로그램과 "세프어레이(SafeArray)" 용어로 포맷된 데이터 매니저(316)간에 데이터가 패스된다. 세프어레이는 마이크로소프트 액셀의 상업적인 스프레드 시트 프로그램과 사용될 수 있는 Microsoft OLE(Object Linking and Embedding Language)의 일부이다.

데이터 매니저(316)는 "콘테이너"로 불리우는 일련의 메모리 구조로서 구현된다. 콘테이너의 포맷은 저장되는 데이터의 타입에 의존한다. 예를 들어, 핀 맵 데이터는 레벨 시트 데이터를 수용하는 콘테이너와 상이한 포맷으로 콘테이너에 저장된다. 콘테이너의 포맷은 저장되어야 하는 정보를 근거로 하여 공지된 프로그래밍 기술을 이용하여 정의된다.

콘테이너가 내장될 수 있다. 예를 들어, 핀 맵용 콘테이너내에, 몇몇 핀 그룹이 존재할 수 있다. 각 핀 그룹내에, 몇몇 핀이 존재할 수 있다. 이러한 예에서, 콘테이너는 3가지 레벨의 네스팅(nesting)을 가지고 있다. 그러나, 임의 레벨의 네스팅이 가능할 수 있다.

C++ STL(Standard Template Library)는 클래스내의 데이터를 조작하는데 유용한 방법을 포함한다. 이러한 방법중 두 가지는 "MAP" 와 "VECTOR"이다. 일반적으로, "콘테이너"는 논리순서보다는 오히려 프로그램에 의해 생성되는 순서로 컴퓨터 메모리에 할당된다. 그러나, 정보를 쉽게 검색하게 위해서, "키" 세트는 메모리내의 물리적 위치와 컨테이너의 이름사이에 생성된다. "키"는 색인처럼 동작하여서, 소망 데이터는 올바른 키를 찾음으로써 발견될 수 있다.

MAP 방법은 2진 트리와 같이 브렌치되는 키 세트를 생성한다. VECTOR 방법은 리스트와 같이 순서 정렬된 색인을 생성한다. MAP 방법의 장점은 랜덤 아이템을 신속하게 찾을 수 있다는 것이다. MAP 방법의 단점은 순서가 정해져 있지 않다는 것이다. 예를 들어, 핀 맵 데이터 시트내의 핀의 순서는 시험 프로그램에서 중요한 정보이다. MAP 방법을 바로 이용하면, 순서가 정해지지 않는다는 것이다. 데이터 매니저(316)에 데이터를 저장하는 방법은 각각의 잇점을 얻기 위해 MAP와 VECTOR STL 방법을 결합하여 사용한다.

기능의 저장 및 검색에 추가로, 데이터 매니저(316)는 필요한 데이터 관리 기능을 수행하기 위한 다른 방법을 포함한다. 추가 기능중 하나는 SELECT DEFAULT기능이다. 위에서 설명된 바와 같이, 데이터의 특정 클래스를 위한 복수의 데이터 세트가 존재할 수 있다. 예를 들어, 하나의 시험 작업내에 복수의 핀 맵이 존재할 수 있다. SELECT DEFAULT 방법은 어느 데이터 세트가 디폴트로서 사용되어야 하는 지를 나타낸다. 예를 들어, 시험 작업이 두 개의 핀 맵, PIN MAP1과 PIN MAP2를 가지고 있다면, PIN MAP1은 디폴트에 설정될 수 있다. 핀 맵 데이터를 검색하는 검색 기능을 사용하여, PIN MAP2가 검색 동작이 수행될 때를 분명하게 나타내지 않으면, PIN MAP1로부터 데이터를 얻을 수 있다.

SELECT ACTIVE 방법은 또한 데이터 매니저(316)를 포함할 수 있다. 그 방법은 SELECT DEFAULT와 유사할 수 있지만, 사용될 수 있는 콘테이너를 명시할 수 있다.

포함될 수 있는 다른 방법은 REPLACE 방법일 수 있다. 위에서 설명된 바와 같이, 데이터 매니저(316)는 특정 디바이스를 시험하는 시험 작업용 데이터를 위한 일련의 컨테이너이다. 시험 작업의 개발 동안 또는 특정 작업의 시험사이에, 일부 디바이스용 데이터를 변경하는 것이 바람직하다. REPLACE 방법으로, 하나의 데이터 세트는 다른 세트로 대체될 수 있다. 예를 들어, 사용자가 시험 작업을 개발하는 공정의 일부로서 디바이스 핀 맵 스프레드 시트를 편집하였다면, 그 핀 맵 데이터 시트와 관련된 매크로는, 시험 작업이 실행될 때 새로운 데이터가 사용될 수 있는 REPLACE 방법을 자동적으로 호출할 수 있다. 데이터가 변경될 때 자동적으로 실행하기 보다는, REPLACE 방법은 사용자 입력에 응답하여 동작하는 매크로에 의해 호출될 수 있다.

데이터 매니저내의 데이터가 디바이스 드라이버(328)중 하나에 의해 변경되면 유사한 방법이 필요하다. 설명된 바와 같이, 데이터 매니저(316)는 액셀 워크북내에서 유지되는 데이터로부터 분리되는 데이터의 사본을 필수적으로 유지한다. 엑셀 워크북내의 변경은 REPLACE 방법에 의해 데이터 매니저에 반영된다. 데이터 매니저내의 데이터가 워크북내에서 변경된 것이 아니라면, REPLICATE 방법이 필요하다. 예를 들어, 디바이스 드라이버가 데이터 매니저내의 하나의 컨테이너에서 값을 변경한다면, 데이터 매니저는 REPLICATE 방법을 호출할 수 있다.

REPLICATE 방법은 데이터 매니저(3616)로부터 현재의 데이터를 역으로 판독하는 엑셀 툴을 호출한다. 데

이터는 적절한 엑셀 스프레드 시트로 로딩되고, 이로 인해, 워크북과 관련된 엑셀 기능은 변경된 데이터에 따른 데이터를 재산출할 수 있다. 그 다음, 변경된 스프레드 시트내의 데이터는 데이터 매니저에서 대체된다. 데이터 매니저에 의해 저장되는 데이터를 변경하는 동작을 수행하는 시험 템플릿은 실행이 종료되었을 때 REPLICATE 방법을 이용한다.

동작

테스터(112)를 제어하는 소프트웨어의 구조가 설명되어 있다. 소프트웨어를 여러번 사용할 수 있다. 하나의 적당한 사용은 소프트웨어가 먼저 시험 작업을 개발한 후, 반도체 디바이스의 처리동안에 그 시험 작업을 반복적으로 실행하도록 사용될 수 있다는 것이다. 하나의 적당한 사용은 디바이스 엔지니어가 필요하다면, 시험 템플릿(320)을 먼저 편집할 수 있다는 것이다. 매우 소수의 시험 템플릿이 여러 종류의 디바이스를 시험할 수 있게 하고자 한다. 그러므로, 시험 템플릿의 편집은 필요하지 않을 것같다.

반도체 시험 디바이스의 개발자가 행할 수 있는 다음 단계는 디바이스 데이터와 플로우 개발 툴(310)(도3)을 사용하는 것일 수 있다. 이러한 툴은 개발하고자 하는 특정 디바이스용 엑셀 워크북을 오픈함으로써 엑세스된다. 특히, 이러한 툴은 도 4내지 도 11에 도시된 스프레드시트를 나타낸다. 매크로스 또는 위저드는 이러한 형식을 통해 사용자를 안내하도록 사용될 수 있다. 특정 시트에 대한 한번의 테이터 입력을 행할 수 있는 타당성검사 매크로스가 완료되어, 다른 테이터와 일치하지 않는 어느 테이터가 검출되고 교정될 수 있다. 예를들면, 프로그래머는 핀 맵상에 없는 특정 핀에 대한 테이터 값을 입력할 수 있다. 핀이 핀맵의 외부에 부주의로 남겨졌든지 또는 특정핀에 대해 잘못된 명칭이 사용되었든지 어느 하나의 이유로 그러한 하나의 입력은 에러를 가리키는 것이다.

바람직한 실시예에서, 소프트웨어는 코드화 되어서, 만약 테이터가 입력되었거나 변경되어서 타당성검사를 하지 않으면 테스트 작업은 실행되지 않을 것이다. 사용자는 주문 메뉴바로부터 타당성검사를 선택할 수 있다.

일단 테이터가 입력되면, 사용자는 "컴파일(compile)" 옵션을 선택한다. 바람직한 실시예에서, 컴파일 옵션은 플로우 테이터 시트와 관련한 주문 메뉴 항목으로부터 선택된다. 컴파일 옵션은 작업 플로우 테이터 구조를 구성하기 위해 플로우 제어 소프트웨어(314)를 트리거 한다. 그것은 또한 테이터 메니저(316)내 방법이 테이터를 액셀 워크북으로부터 테이터 메니저에 전달하도록 한다.

개발 환경에서, 테스트 엔지니어는 프로그램 디버깅 처리를 개시한다. 어떻게 해서든지 엔지니어는 디바이스를 테스터(112)에 접속할 수 있다. 대안으로 테스트 엔지니어는 시뮬레이터를 사용하여 테스트 프로그램을 디버깅 할 수 있다. 시뮬이터는 데스터에 접속된 디바이스를 갖는 테스터(112)의 동작을 시뮬레이트 하는 소프트웨어 프로그램이다.

엔지니어는 프로그램을 통해 나아가기 위해 상술한 디버깅 툴을 사용한다. 테스트 엔지니어가 프로그램내에서 에러를 발견했을때, 엔지니어는 테스트 템플릿을 편집할 수 있다. 테스트 템플릿이 인터프리터형 언어인 비주얼 베이직으로 쓰여졌기 때문에, 테스트 작업을 재 컴파일 하지 않고, 변경이 즉시 효과적으로 일어날 수 있다.

대안으로서, 테스트 엔지니어는 하나의 디바이스 테이터 구조내 일부 데이터를 변경할 수 있다. 데스트 엔지니어는 도4 내지 도11에 도시된 스프레드시트 스크린을 통해 데이터를 액세스 할 수 있다. 그리고 나서 타당성 검사 과정이 다시 개시된다. 타당성 검사의 한 과정으로서, REPLACE 툴은 자동적으로 실행된다. REPLACE 툴을 자동적으로 사용하는 것은 데이터 매니저(316)가 날짜 데이터에 제공하는 것을 보장한다.

만약, 디버깅 하는 부분으로서, 테스트 엔지니어가 프로그램 플로우를 변경하면, 플로우는 새로운 작업 플로우 구조가 세워지도록 재 컴파일 되어야만 한다.

일단 요구된 변경은 디바이스 날짜, 테스트 테플릿 또는 디바이스 플로우에 만들어지고, 테스트 작업은 다시 실행될 수 있다. 테스트 프로그램에 변경을 행하기 위해 요구된 시간은 매우 짧을 수 있다. 어떤 종래 기술의 시스템에서, 아주작은 변경일지라도, 테스트 작업은 완전한 재 컴파일을 요구한다. 그 결과 어떤 변경은 한 시간에 이른다. 본 발명에서, 템플릿에 대한 변경은 어떤 재 컴파일도 요구하지 않는다. 디바이스 데이터에 대한 변경은 오로지 변경된 데이터가 재적재 되도록 요구한다. 만약 변경되지 않으면 오직 프로그램 플로우가 컴파일 되어야 한다. 왜냐하면 컴파일이 모든 데이터가 아닌 오직 플로우와 함께 처리되기 때문에, 부차적인 사정이 있는 경우에 실행될 수 있기 때문이다.

전형적 개발 세팅에 있어서, 테스트 엔지니어는 테스트 작업이 기대치에 이르기 까지 테스트 작업의 변경과정을 반복적으로 되풀이 할 수 있다. 일단 완성되면, 테스트 작업은 제품세팅에 전달될 수 있다. 제품 세팅에 있어서, 디바이스 데이터 및 플로우 개발 툴은 사용되지 않는다. 유사하게, 주문 인터페이스가 개발된다. 주문 인터페이스는 단지 테스터를 동작하는데 필요한 제어를 일부 테스트에 제공하고, 테스팅이 처리중임을 인식하고 있는 운영자에게 단지 기초적인 출력정보를 보여주도록 표시한다. 비록 테스터가 많은 데이터를 측정하고 저장할 수 있을지라도, 제품환경내에서 운영자에게 아주 소량의 데이터만이 제시된다.

소프트웨어가 액셀을 사용하여 실행되는 곳에서는, 비주얼베이직 프로그래밍 언어는 주문 제품 인터페이스를 준비하는데 용이하다. 인터페이스내에서 유사한 것은 테스팅을 시작하고 멈추도록 하는 제어이고, 확실한 테스트를 하지 못하는 디바이스 수를 지시하는 표시부이다.

다른 실시예

일 실시예에 기재되었음에도 불구하고, 수많은 대체 실시예 또는 변화가 가능하다. 예를들면, 본 발명은 몸체 및 개별의 컴퓨터화된 워크스테이션을 갖는 테스터와 함께 기재되어 있다. 컴퓨터화된 워크스테이션은 테스터 몸체로부터 물리적으로 분리될 필요가 없다. 예를들면, 그것은 테스터 몸체내에 구성될 수 있기 때문이다. 다른 구성도 가능하다. 예를들어, 어떤 테스터는 주 프레임과 테스트 헤드를 갖는다. 본 발명을 설명할 목적으로 상기 테스터는 하나 또는 몇몇 다른 패키지로 구성되어 있다.

상세 데이터 및 플로우 시트의 예는 주어져 있다. 각 행은 각 렬이 플로우내 항목 또는 단계에 대한 일정한 유형의 정보를 포함하는, 특정수의 렬과 함께 도시되어 있다. 작동되는 유형에 따라, 테스터의 하드웨어 또는 디바이스의 요구사항은 테스트 되고, 각 로우에 포함된 정보의 수 및 종류는 다를 수 있다. 다소간의 렬이 이용가능 하다. 스프레드시트의 각 유형에 대한 가능 렬의 여분의 세트를 갖는 것이 가능하다. 사용자는 특정 작업에 중요한 렬을 골라 잡는다. 위저드는 선택과정을 통하여 프로그래머를 유도하도록 한다. 위저드는 사용자로 하여금 선택하도록 하는 매크로 이다. 여분의 세트내에서 특정 스크린상에 표시하기 위해 선택되지 않은 그러한 렬은, 스프레드시트로 부터 완전히 생략되거나, 제시될 수 있지만 표시되지는 않는다. 제시되었으나 표시되지 않은 렬은 무시되거나 또는 디폴트 값으로 주어진다.

우리는 C++ 내에 쓰여진 플로우 제어 소프트웨어를 기술하였다. C++은 상업적으로 이용가능한 스프레드시트 프로그램인 액셀의 일부로서 프로그램 언어인 비주얼 베이직보다 더욱 신속히 실행될 수 있다고 널리 알려져 있다. 그러나, 만약 C++ 내에 쓰여져서, 비록 플로우 제어 소프트웨어가 더 빠르다 할지라도, 디바이스 및 액셀 프로그램을 사용하여 플로우 데이터 입력에 인터페이스 하는 시간은 더 느리다. 그래서 어떤 경우, 플로우 제어 소프트웨어는 비주얼 베이직 또는 상업적으로 이용가능한 스프레드시트 프로그램의 일부인 다른 매크로 프로그램 언어로 구현하는 것이 바람직 할 것이다.

플로우 제어 소프트웨어(314)가 단일 부분을 테스트하는 테스트 작업을 제어하는 것이 도 13에 도시되어 있다. 아마 테스터(112)는 수백 내지 수천개의 핀을 갖고 있다. 그러므로 테스터(112)에 동시에 많은 부분을 접속하기에 충분한 핀이 존재하는 것이 가능하다. 이 경우, 핀은 "사이트(site)"로 그룹지어져 있다. 각 사이트내 핀은 동일 데이터를 갖고, 동일 측정치를 만든다. 그러나, 만약 한 사이트가 에러를 발견하면, 다른 처리가 그 사이트상에 요구된다. 그러므로 플로우 제어 소프트웨어(314)는 다중 사이트상에서 동시에 테스트를 행할 수 있는 구조로 되어 있다. 그러나, 어느 한 사이트가 잘못되면 그 사이트는 분리되어 테스트 될 수 있다. 이러한 기능을 실현하기 위해, 단계(1326)는 사이트의 상태를 개별적으로 트랙하기 위해 변경될 수 있다. 그래서 각 사이트에서 디바이스가 테스트 작업에서 통과 또는 잘못된 어떤 테스트를 갖는 것에 따라, 단계(1328)는 각 사이트에 대한 템플릿을 선택하기 위해 변경될 수 있다.

또한, 테스트 시스템이 반도체 컴포넌트를 테스트 하는 것이 도시되어 있다. 유사한 소프트웨어는 인쇄회로기판 또는 다른 디바이스에 대한 테스트 시스템을 제어하기 위해 사용될 수 있다.

추가로, 상업적으로 이용가능한 스프레드시트 프로그램의 사용이 소프트웨어로 하여금, 사용자가 쉽게 이해할 수 있는 방법으로 신속히 실행될 수 있는 테스트 시스템을 제어가능 하도록 하는 것이 기술되었다. 도14는 실행될 수 있는 다른 툴의 실시예를 도시하고 있다. 도14는 디버그 또는 분석 툴로서 주로 사용되는 툴을 도시하고 있다. 여기서 데이터 입력 스크린이 도시되어 있다. 상기 스크린은 스프레드시트 보다는 비주얼 베이직을 사용하여 대화상자로서 구현된다. 도14의 특정 예는 단일 핀(124,도1)의 하드웨어 셋업에 관련된다. 대화상자는 데이터가 입력될 수 있는 몇몇 필드를 포함하고 있다. 그 데이터는 테스터내 하드웨어에 통과된 값을 표시한다. 이 대화상자가 사용될때, 그것은 플로우 제어(314)를 통하여 표준 데이터 경로로 바이패스 하고, 테스터 하드웨어를 액세스 하기 위해 직접 디바이스 드라이버(328)를 호출한다.

가능한 변화의 다른 예로서, 상업적으로 이용가능한 스프레드시트 프로그램은 액셀 또는 사용된 그의 픽처와 다른 액셀의 버전과는 다른 것일 수 있다. 도 15는 액셀'97로 실행된 데이터 입력 스크린의 예를 보이고 있고, 상업적으로 이용가능한 스프레드시트 프로그램은 마이크로소프트사가 판매하였다. 몇가지 추가적 특징이 액셀'97을 통하여 가능하다. 제어요소(1510)는 그룹표시 제어이다. 데이터의 다수 행이 동일 그룹내에 있을 때, 표시부는 붕괴되어서 오직 단일 행만이 그 그룹에 표시될 수 있다. 예를들면, 도15에서 (1512) 및 (1514)로 지시된 행은 모두 "rdcycle" 라고 명명된 시간설정내에 있다. 모든 행이 도시되고, 그룹이 확장되었다. 그러나, 그룹표시제어(1510)가 사용된다면, 단지 단일 행만이 그 그룹에 대해 표시될 수 있다. 비록 다른 행들이 보이지 않더라도, 그것들은 도15의 데이터 입력 스크린을 사용하여 실행된 어떤 작동에 의해 여전히 행해질 수 있다.

액셀'97의 사용을 통해 이용될 수 있는 다른 특징은 그 유형에 기초하여 달리 보이는 데이터를 만든다. 예를들면, 셀(1516)내에 있는 데이터는 셀(1518)내에 있는 데이터와 다른 모습을 갖는다. 그것은 셀(1516)내에 있는 데이터가 어드레스 버스에 대해 한 유형을 할당하고, 셀(1518)내에 있는 데이터가 데이터 버스에 대해 한 유형을 할당하고 있기 때문이다. 시각적으로 구별되는 다른 유형의 데이터를 만드는 것은 소프트웨어를 보다 쉽게 이용하기 위해서 이다.

도 16a 및 도 16b는 다른 인핸스먼트를 도시하고 있다. 그룹제어요소(1610)는 확장된 상태로 보여지고 있다. 그래서, "General Info"로 가리키는 그룹에 대한 요소의 목록이 있다. 대조적으로, 그룹제어(1612)는 비확장 상태로 보이고 있다. 그래서, "Digital Setup"그룹에 대해 어떤 입력도 없다.

도 16a 및 도 16b는 또한 시각적으로 구별되어 나타나는 그룹제조의 잇점을 강조한다. 예를들면, 필드(1614) 및 (1616)은 다른 유형의 데이터를 포함하고 있기 때문에 다르게 나타난다.

도 16a 및 도 16b에 도시된 다른 특징은 네비게이션 특징이다. 상기 실시예에서, 툴은 스크린 밑에 있는 선택탭으로 액세스 된다. 도 16a 및 도 16b는 스크린이 "hot link"와 함께 링크될 수 있음을 설명한다. 다른 스크린으로부터 스크린 이름에 클릭을 함으로써, 사용자는 스크린 표시부를 변경할 수 있다. 또한 네비게이션 툴 바는 인터넷 네비게이터에서 발견될 수 있는 툴을 포함하고 있다. 툴은 작동을 이전 스크린에 이동하도록 또는 이전 특정 애용 스크린에 가도록 실행된다.

액셀'97을 사용하는 다른 잇점은 그것이 객체 지향 언어라는 것이다. 각 데이터 스크린이 목적이 아니다. 데이터 스크린과 같이 각 목적은 그 목적에 관한 클래스를 갖는다. 모든 데이터 시트는 어느 데이터 시트에 공통되는 동작을 한정하는 베이스 클래스로 주어질 수 있다. 예를들면, 에러 핸들링, 데이터 유형 및 포맷팅은 베이스 클래스에서 정의될 수 있다. 유효 데이터 및 데이터 매니저로 데이터 전송과 같이 모든 데이터 시트로서 처리되는 다른 기능들은, 베이스 클래스에 대해 "public methods" 로 정의될 수 있다. 이 방법에 있어서, 이러한 모든 기능들이 비록 단 한번만 상술되었지라도, 모든 데이터 스크린에 대해 제공된다. 마찬가지로 중요한 각 디바이스 데이터 또는 플로우 데이터 툴은 동일 방법으로 작동

한다.

객체 지향 프로그램밍을 이용하여 가능한 또다른 변화는 인터-시트 에러의 순시 타당성검사를 받아야 한다. 타당성검사 룰은 상술될 수 있고, 입력된 바와 같이 데이터는 체크된다. 타당성검사 루틴은 여전히 실행중인 다중 시트 사이의 불일치 데이터를 체크한다.

그러므로, 본 발명은 부가된 청구항의 사상 및 범위에 의해서만 한정되어야 한다.

(57) 청구의 범위

청구항 1

복수의 테스트 신호를 생성하고 측정하는 전자 회로를 갖는 몸체 및 제어정보를 몸체내의 전자회로에 제공하는 컴퓨터 워크스테이션을 가지며, 추가로 컴퓨터 워크스테이션내의 제어 소프트웨어를 가지는 형태의 반도체 테스트 시스템에 있어서,

상기 제어 소프트웨어는:

a) 테스트중인 디바이스에 관한 데이터를 갖도록 구성된 복수 형태의 스프레드 시트의 제 1 부분 및 반도체 디바이스상에서 테스트 작업동안에 수행되는 단계에 영향을 미치는 정보를 갖도록 구성된 복수 형태의 스프레드 시트의 제 2 부분을 구비한 복수 형태의 스프레드 시트를 가지는 상업적으로 이용가능한 스프레드 시트 프로그램;

b) 각각이 몸체내의 전자회로중의 일부분을 제어하는 복수의 디바이스 드라이버 프로그램 요소; 및

c) 복수 형태의 스프레드시트의 제 2 부분으로부터 데이터를 수신하여, 제 2 형태의 스프레드 시트에 있는 데이터에 의해 지시된 순서대로 제 1 형태의 스프레드 시트로부터 복수의 프로그램 요소중 선택된 것에 데이터를 제공함으로써 테스트 작업의 실행을 제어하는 프로그램 수단;을 포함하는 것을 특징으로 하는 테스트 시스템.

청구항 2

제 1 항에 있어서, 상기 상업적으로 이용가능한 스프레드 시트 프로그램은 스프레드 시트로부터 데이터를 판독하도록 집적화된 집적 프로그래밍 언어를 포함하고, 상기 프로그램 수단은 집적 프로그래밍 언어로 쓰여진 플로 제어 소프트웨어를 포함하는 것을 특징으로 하는 테스트 시스템.

청구항 3

제 2 항에 있어서, 상기 집적 프로그래밍 언어는 Visual Basic인 것을 특징으로 하는 테스트 시스템.

청구항 4

제 1 항에 있어서, 상기 복수 형태의 스프레드 시트는 워크북내에 편성되고, 상기 워크북에서의 1 스프레드 시트에서의 데이터는 상기 워크북에서의 다른 스프레드 시트로부터 액세스가능한 것을 특징으로 하는 테스트 시스템.

청구항 5

제 1 항에 있어서, 상기 상업적으로 이용가능한 스프레드 시트 프로그램은 Excel인 것을 특징으로 하는 테스트 시스템.

청구항 6

제 1 항에 있어서, 각각이 테스트 작업의 1 단계동안에 수행되는 기능을 나타내는 복수의 테스트 프로그램 요소를 추가로 포함하고, 여기에서 복수의 스프레드 시트의 제 2 부분에 있는 스프레드 시트는 복수의 행을 포함하는 플로 시트이고, 상기 적어도 복수의 행의 일부분 각각은:

a) 테스트 프로그램 요소에 대한 기준; 및

b) 수행될 때, 기준된 테스트가 실패를 지시한다면, 테스트중인 디바이스가 할당되도록 분류를 지시;를 포함하는 것을 특징으로 하는 테스트 시스템.

청구항 7

제 6 항에 있어서, 플로 시트에 있는 적어도 복수의 행의 일부분의 각각은:

a) 기준된 테스트가 실패할 때 세트되어지는 플래그의 네임;

b) 일 행에 있는 논리식이 다른 행에 포함된 플래그의 명칭을 포함하며, 기준된 테스트가 수행되는지를 지시하는 논리식을 나타내는 셀을 추가로 포함하는 것을 특징으로 하는 테스트 시스템.

청구항 8

제 7 항에 있어서, 상기 각 플로 시트는 추가로 테스트 작업의 실행을 변경하는 인수를 입력하는 수단과 관계되어 있는 것을 특징으로 하는 테스트 시스템.

청구항 9

제 6 항에 있어서, 각각의 테스트 프로그램 요소는 테스트 템플릿을 포함하고, 복수의 스프레드 시트의 제 2 부분에서의 스프레드 시트는 복수의 행을 포함하는 인스턴스 시트이며, 테스트 템플릿의 인스턴스를 정의하는 적어도 복수의 행의 일부분 각각은:

a) 테스트 템플릿에 대한 기준;

b) 테스트 인스턴스에 대한 식별자;를 포함하고,

상기 인스턴스 시트는 추가로 테스트 인스턴스를 형성하기 위해 테스트 템플릿과 결합되도록 테스트중인 디바이스에 관한 데이터를 갖도록 구성된 스프레드 시트중 선택된 것을 특정화하는 수단을 포함하고, 여기에서, 플로 시트에 있는 테스트 프로그램 요소에 대한 기준은 테스트 인스턴스에 대한 식별자를 포함하는 것을 특징으로 하는 테스트 시스템.

청구항 10

제 9 항에 있어서, 각각의 테스트 템플릿은 해석 프로그래밍 언어로 기록되는 것을 특징으로 하는 테스트 시스템.

청구항 11

제 1 항에 있어서, 상기 프로그램 수단은:

a) 복수 형태의 스프레드 시트중의 제 2 부분에 있는 스프레드 시트에서의 정보에 의해 결정된 순서로 디바이스 드라이버 프로그램 요소중 선택된 것을 호출하는 플로 제어 프로그램 수단;

b) 복수 형태의 스프레드 시트의 제 1 부분에 있는 스프레드 시트에서의 정보에 의해 결정된 값을 복귀하고, 디바이스 드라이버 프로그램 요소에 의해 호출되는 데이터 검색 프로그램 수단;을 포함하는 것을 특징으로 하는 반도체 테스트 시스템.

청구항 12

반도체 디바이스를 테스트하는 자동 테스트 장치를 제어하는 소프트웨어에 있어서,

복수의 데이터 입력 스크린을 포함하고,

각각의 스크린은:

a) 행, 및 각각이 공통 타입의 데이터를 갖는 열에 편성된 복수의 데이터 셀을 가지며, 테스트해야 할 반도체 디바이스상의 데이터를 갖는 스크린의 제 1 부분 및 테스트해야 할 반도체 디바이스를 위한 테스트 프로그램 실행 플로상의 데이터를 갖는 스크린의 제 2 부분에 있는 데이터 셀 필드;

b) 복수의 툴 아이콘을 갖는 툴 바 필드;

c) 각각이 다른 데이터 입력 스크린을 액세스하도록 선택가능한 탭 필드; 및

d) 그 위에 복수의 메뉴 바 필드를 가지며, 각각의 메뉴 아이템은 복수의 서브-메뉴 아이템을 나타내도록 선택가능하고, 적어도 하나의 메뉴 아이템아래의 서브-메뉴 아이템은 선택된 탭 필드에 응답하여 선택되는 메뉴 바 필드;를 포함하는 것을 특징으로 하는 자동 테스트 장치를 제어하는 소프트웨어.

청구항 13

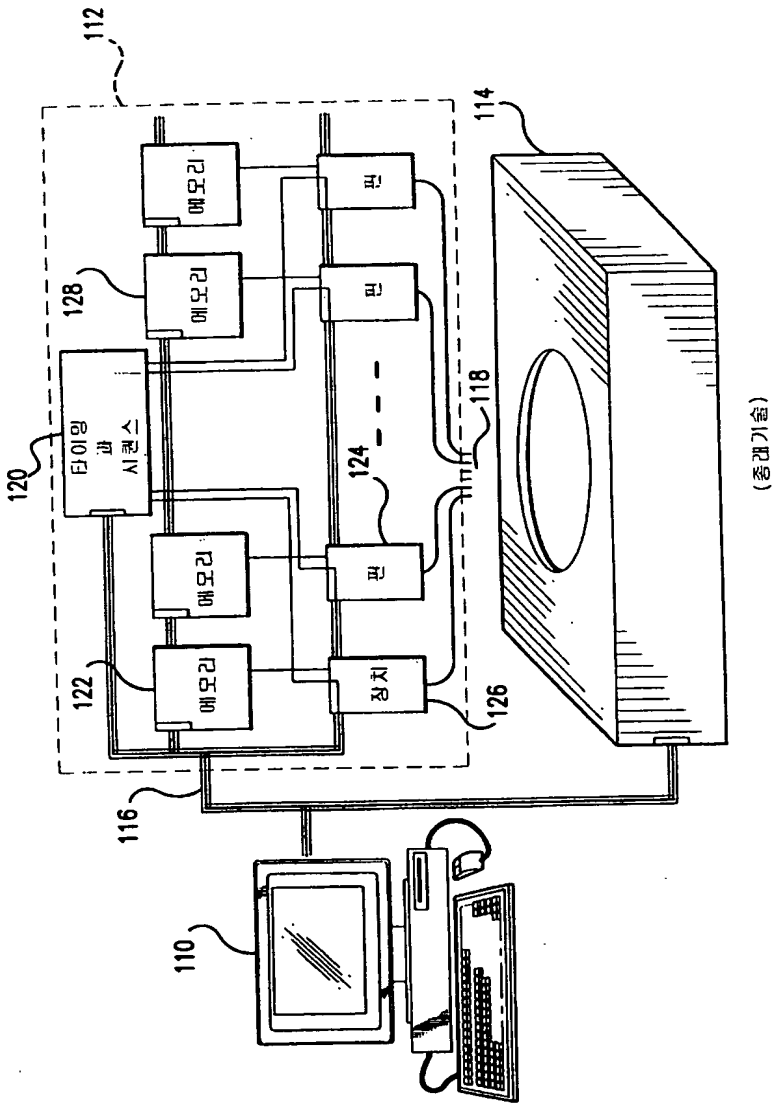
제 12 항에 있어서, 소프트웨어는 상업적으로 이용가능한 스프레드 시트 프로그램에서의 응용으로서 구현되는 것을 특징으로 하는 자동 테스트 장치를 제어하는 소프트웨어.

청구항 14

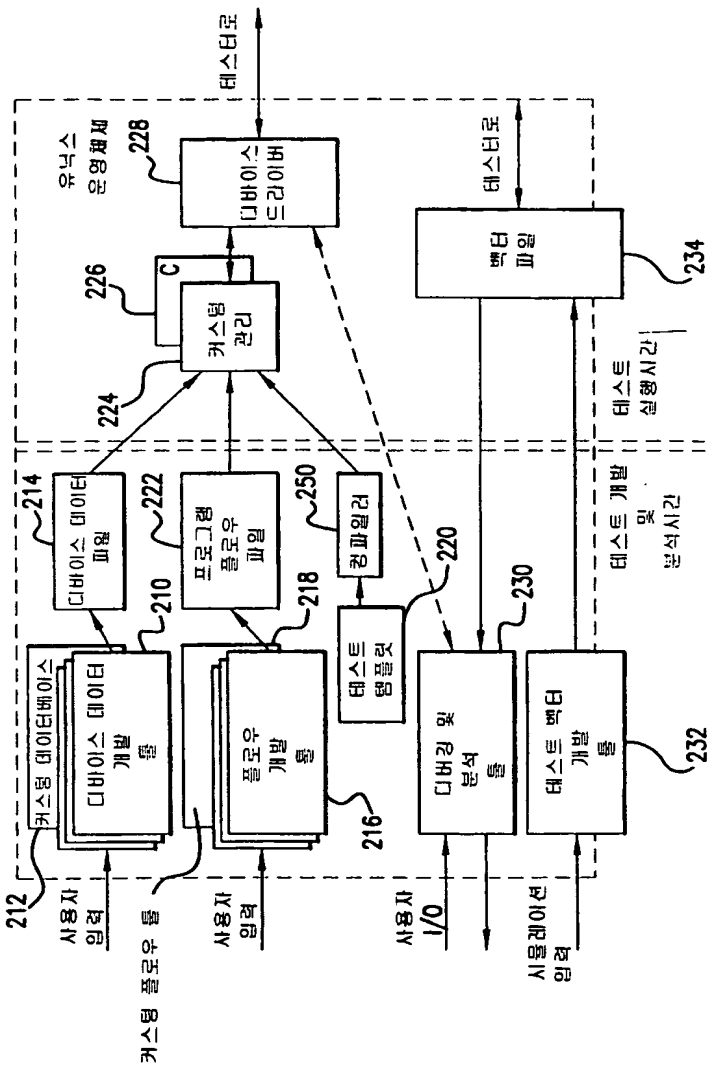
제 13 항에 있어서, 상업적으로 이용가능한 스프레드 시트 프로그램이 Excel인 것을 특징으로 하는 자동 테스트 장치를 제어하는 소프트웨어.

도면

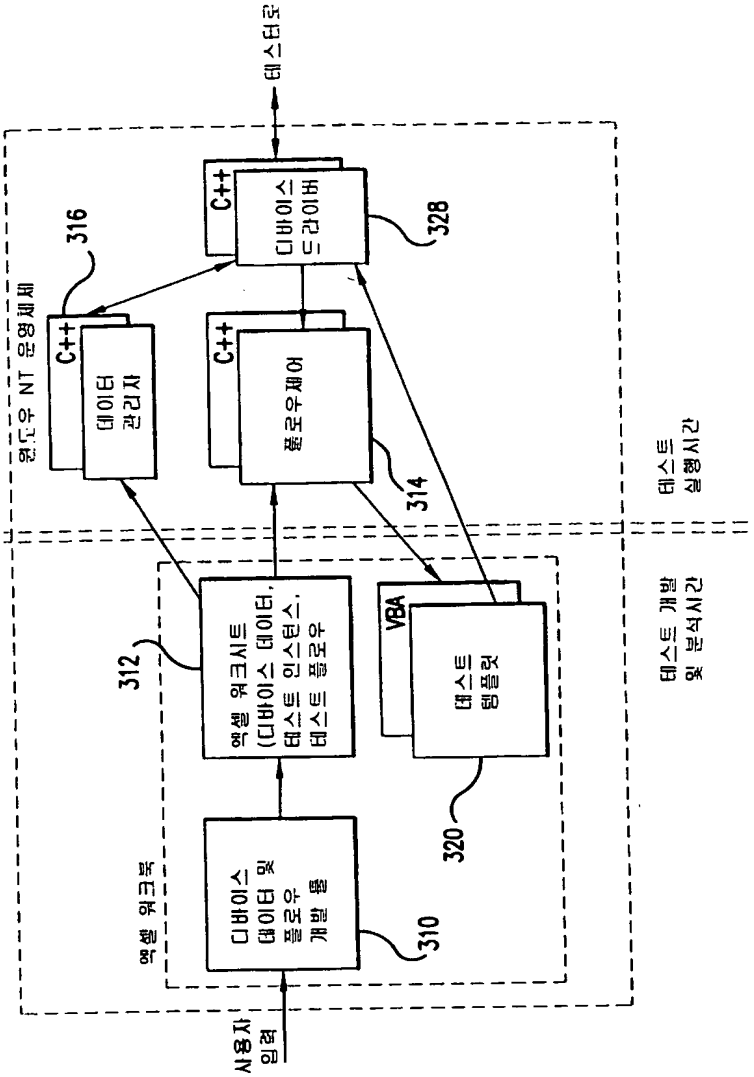
도면1


112
114
메모리
메모리
메모리
메모리
핀
핀
핀
장치
128
120
122
124
126
118
116
110
타이밍 과 시퀀스
(종래기술)

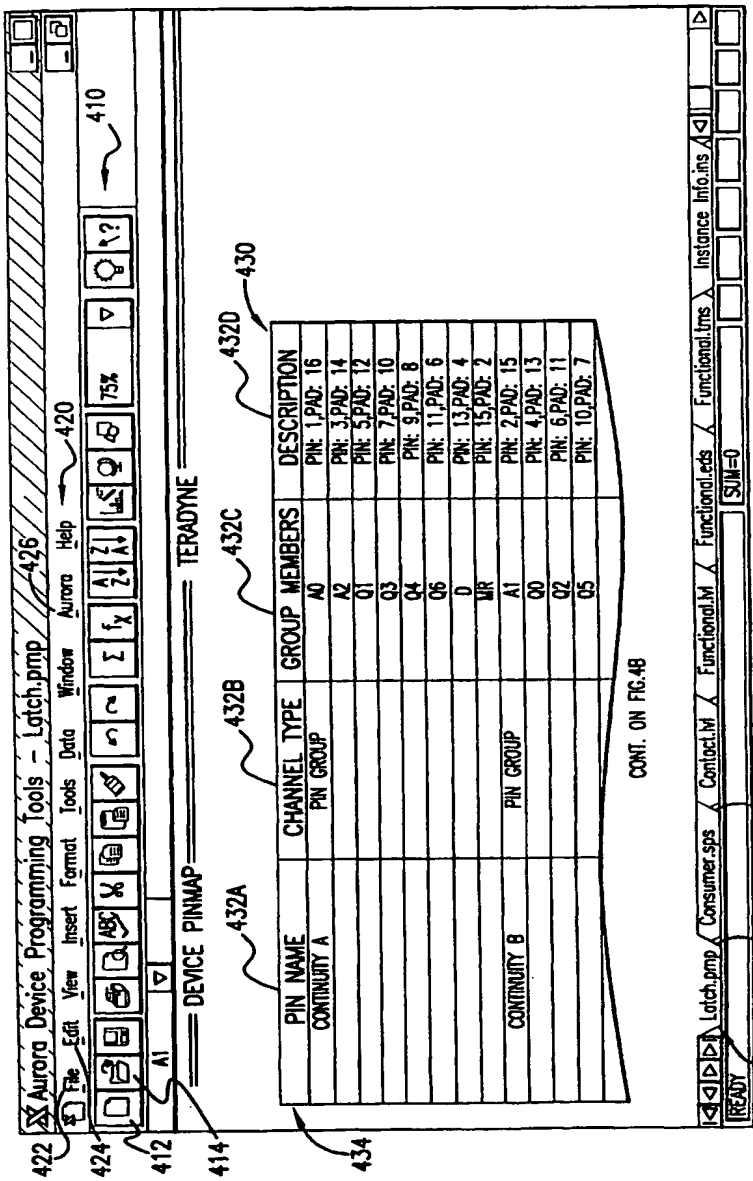
도면2


테스터로
유닉스 운영체제
228
디바이스 드라이버
테스터로
226
C
커스텀 관리
백터 파일
234
224
테스트 실행시간
214
디바이스 데이터 파일
222
프로그램 플로우 파일
250
컴파일러
테스트 개발 및 분석시간
210
218
220
230
커스텀 데이터베이스
디바이스 데이터 개발 툴
플로우 개발 툴
테스트 템플릿
디버깅 및 분석 툴
테스트 벡터 개발 툴
232
216
212
사용자 입력
사용자 입력
사용자 I/O
시뮬레이션 입력
커스텀 플로우 툴
(종래기술)

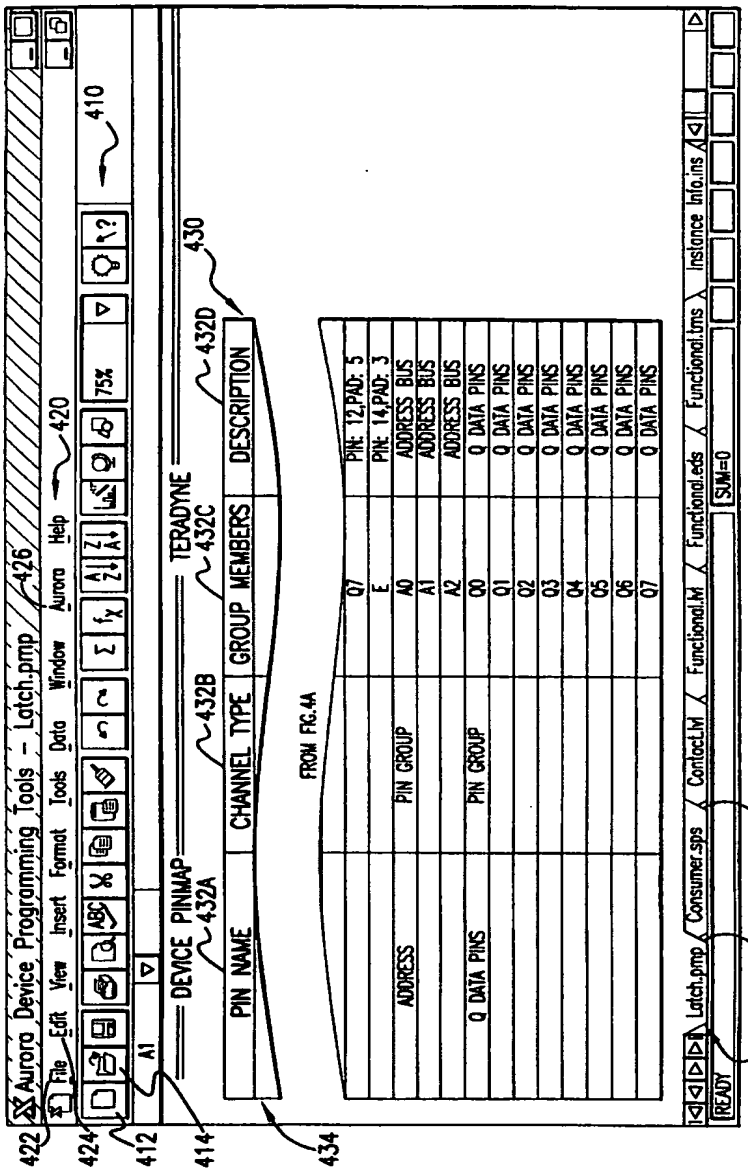
도면3


테스터로
C++
디바이스 드라이버
328
316
C++
데이터 관리자
C++
폴로우제어
314
윈도우 NT 운영체제
테스트 실행시간
테스트 개발 및 분석시간
312
엑셀 워크시트 (디바이스 데이터, 테스트 인스턴스, 테스트 플로우
VBA
테스트 템플릿
320
엑셀 워크북
디바이스 데이터 및 플로우 개발 툴
310
사용자 입력

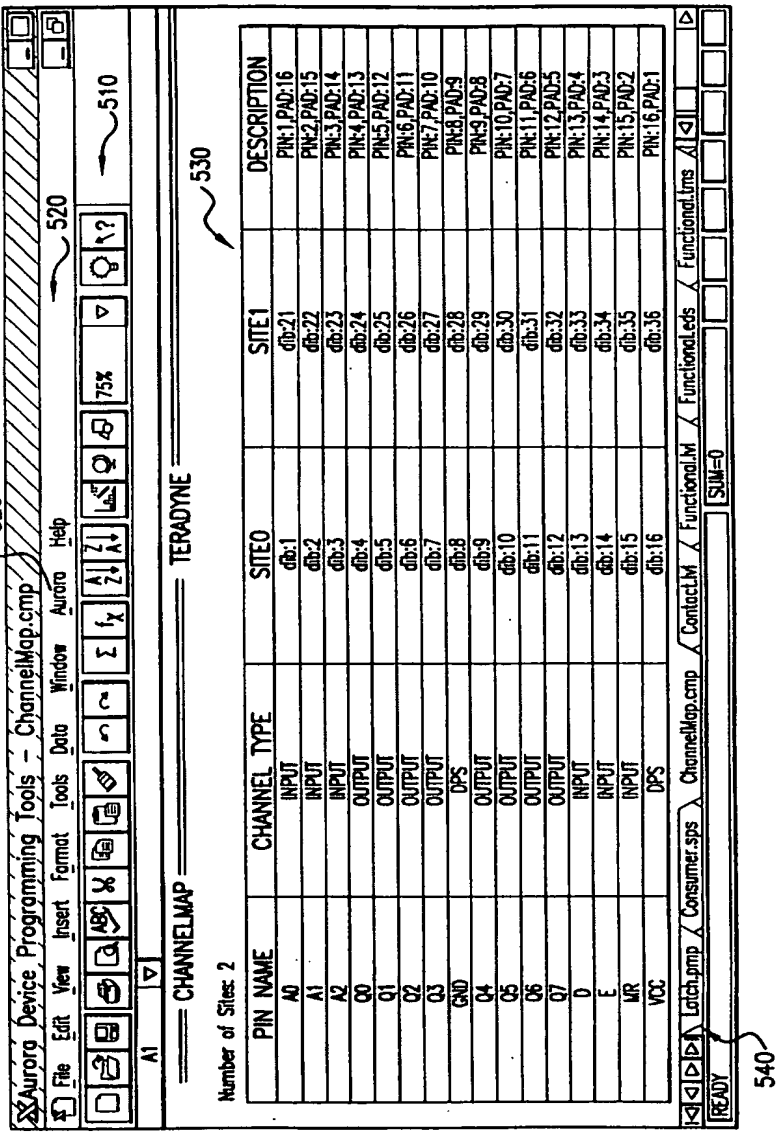
도면4a


Aurora Device Programming Tools - Latch.pmp
File Edit View Insert Format Tools Data Window Aurora Help
A1
DEVICE PINMAP
TERADYNE
PIN NAME
CHANNEL TYPE
GROUP MEMBERS
DESCRIPTION
CONTINUITY A
PIN GROUP
A0
PIN: 1,PAD: 16
A2
PIN: 3,PAD: 14
Q1
PIN: 5,PAD: 12
Q3
PIN: 7,PAD: 10
Q4
PIN: 9,PAD: 8
Q6
PIN: 11,PAD: 6
D
PIN: 13,PAD: 4
WR
PIN: 15,PAD: 2
CONTINUITY B
PIN GROUP
A1
PIN: 2,PAD: 15
Q0
PIN: 4,PAD: 13
Q2
PIN: 6,PAD: 11
Q5
PIN: 10,PAD: 7
CONT. ON FIG.4B
Latch.pmp
Consumer.sps
Contact.M
Functional.M
Functional.eds
Functional.tms
Instance Info.ins
READY
SUM=0
410
420
422
424
426
412
414
430
432A
432B
432C
432D
434
440
442
444

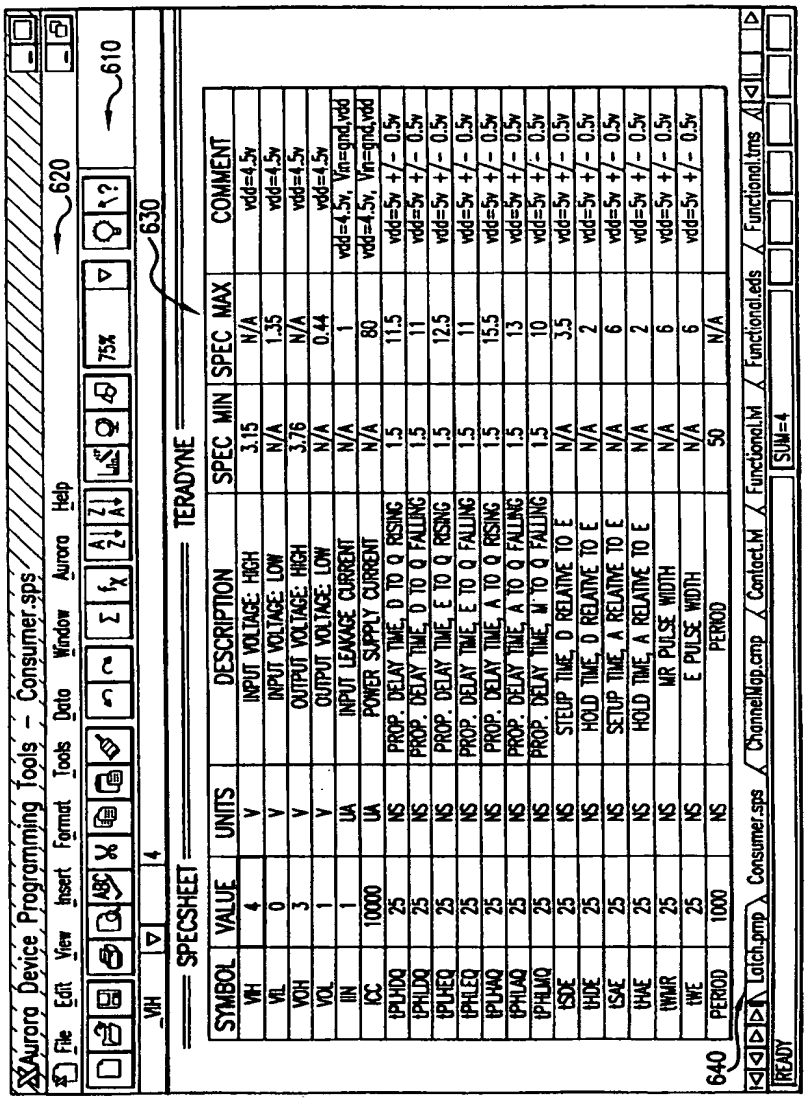
도면4b


Aurora Device Programming Tools - Latch.pmp
File Edit View Insert Format Tools Data Window Aurora Help
410
420
426
422
424
412
414
DEVICE PINMAP
TERADYNE
432A
432B
432C
432D
430
PIN NAME
CHANNEL TYPE
GROUP MEMBERS
DESCRIPTION
FROM FIG.4A
434
ADDRESS
PIN GROUP
A0
A1
A2
ADDRESS BUS
Q DATA PINS
Q0
Q1
Q2
Q3
Q4
Q5
Q6
Q7
Q DATA PINS
E
PIN: 12,PAD: 5
PIN: 14,PAD: 3
READY
Latch.pmp
Consumer.sps
ContactLM
Functional.M
Functional.eds
SUM=0
Functional.tms
Instance Info.ins
440
442
444

도면5

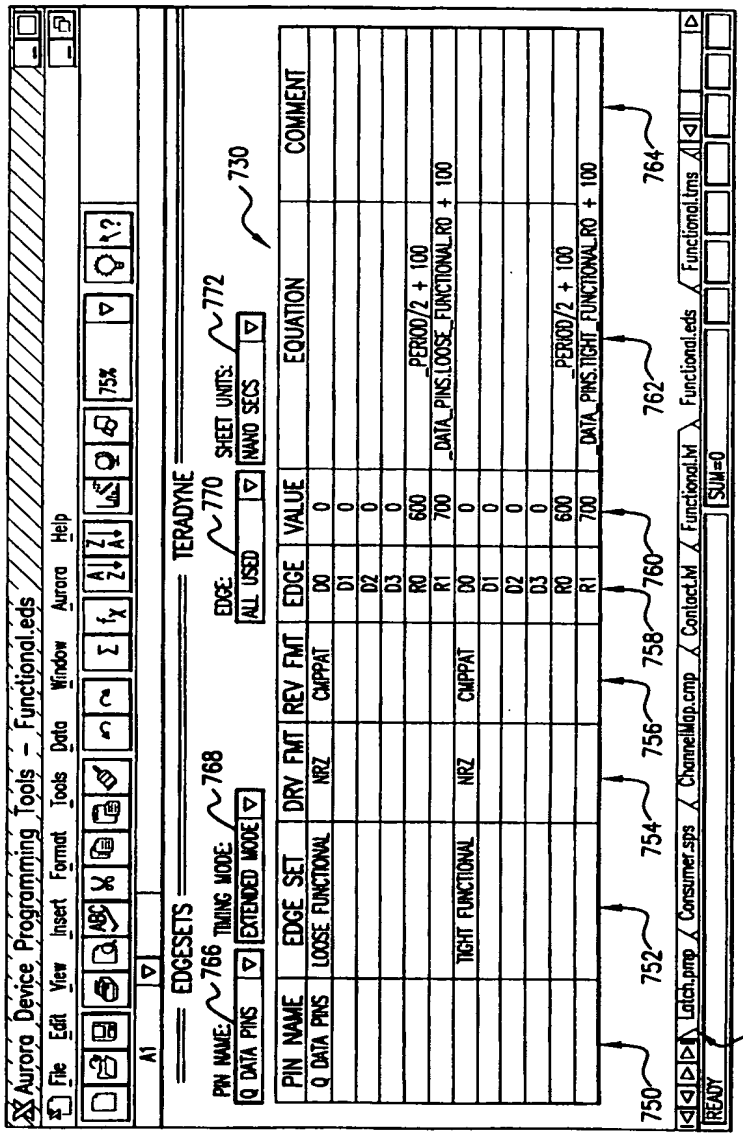

526
Aurora Device Programming Tools - ChannelMap.cmp
File Edit View Insert Format Tools Data Window Aurora Help
520
510
A1
CHANNELMAP
TERADYNE
530
Number of Sites: 2
PIN_NAME | CHANNEL_TYPE | SITE0 | SITE1 | DESCRIPTION
A0 | INPUT | dib:1 | dib:21 | PIN:1,PAD:16
A1 | INPUT | dib:2 | dib:22 | PIN:2,PAD:15
A2 | INPUT | dib:3 | dib:23 | PIN:3,PAD:14
Q0 | OUTPUT | dib:4 | dib:24 | PIN:4,PAD:13
Q1 | OUTPUT | dib:5 | dib:25 | PIN:5,PAD:12
Q2 | OUTPUT | dib:6 | dib:26 | PIN:6,PAD:11
Q3 | OUTPUT | dib:7 | dib:27 | PIN:7,PAD:10
GND | DPS | dib:8 | dib:28 | PIN:8,PAD:9
Q4 | OUTPUT | dib:9 | dib:29 | PIN:9,PAD:8
Q5 | OUTPUT | dib:10 | dib:30 | PIN:10,PAD:7
Q6 | OUTPUT | dib:11 | dib:31 | PIN:11,PAD:6
Q7 | OUTPUT | dib:12 | dib:32 | PIN:12,PAD:5
D | INPUT | dib:13 | dib:33 | PIN:13,PAD:4
E | INPUT | dib:14 | dib:34 | PIN:14,PAD:3
MR | INPUT | dib:15 | dib:35 | PIN:15,PAD:2
VCC | DPS | dib:16 | dib:36 | PIN:16,PAD:1
Latch.pmp Consumer.sps ChannelMap.cmp Contact.M Functional.M Functional.eds Functional.tms
READY
SUM=0
540

도면6

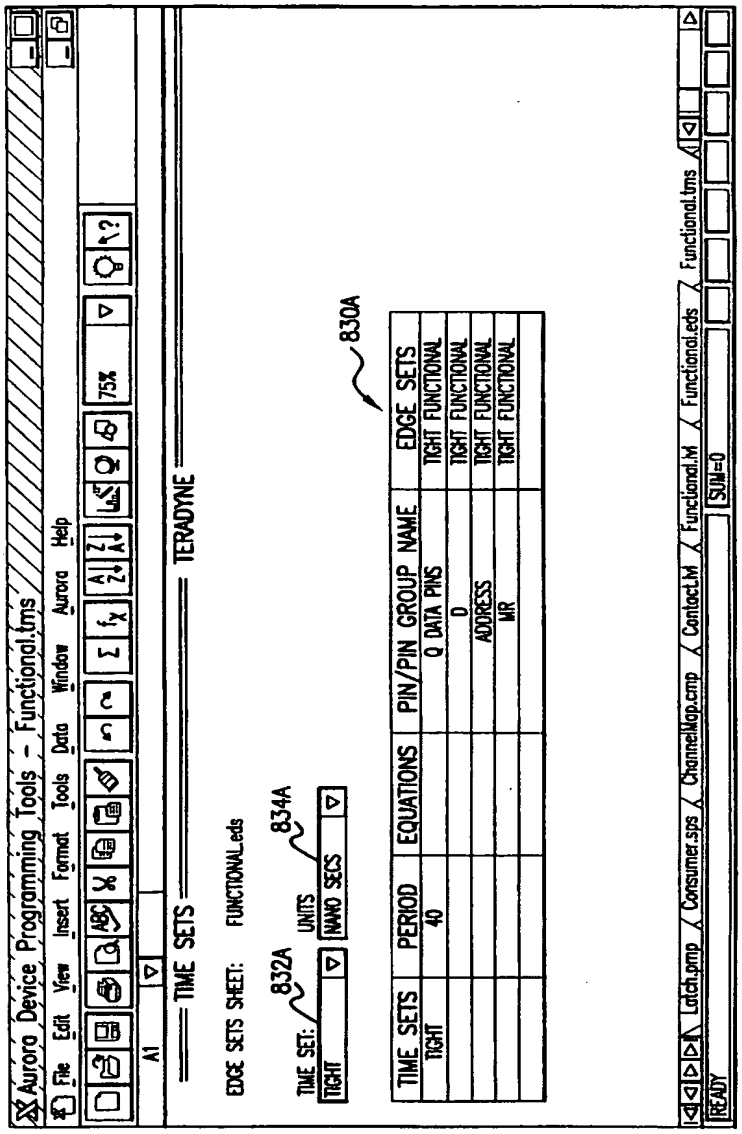


SYMBOL	VALUE	UNITS	DESCRIPTION	SPEC MIN	SPEC MAX	COMMENT
VIH	4	V	INPUT VOLTAGE: HIGH	3.15	N/A	vdd=4.5v
VIL	0	V	INPUT VOLTAGE: LOW	N/A	1.35	vdd=4.5v
VOH	3	V	OUTPUT VOLTAGE: HIGH	3.76	N/A	vdd=4.5v
VOL	1	V	OUTPUT VOLTAGE: LOW	N/A	0.44	vdd=4.5v
IIN	1	UA	INPUT LEAKAGE CURRENT	N/A	1	vdd=4.5v, Vin=gnd,vdd
ICC	10000	UA	POWER SUPPLY CURRENT	N/A	80	vdd=4.5v, Vin=gnd,vdd
tPLHDQ	25	NS	PROP. DELAY TIME, D TO Q RISING	1.5	11.5	vdd=5v +/- 0.5v
tPHLDQ	25	NS	PROP. DELAY TIME, D TO Q FALLING	1.5	11	vdd=5v +/- 0.5v
tPLHEQ	25	NS	PROP. DELAY TIME, E TO Q RISING	1.5	12.5	vdd=5v +/- 0.5v
tPHLEQ	25	NS	PROP. DELAY TIME, E TO Q FALLING	1.5	11	vdd=5v +/- 0.5v
tPLHAQ	25	NS	PROP. DELAY TIME, A TO Q RISING	1.5	15.5	vdd=5v +/- 0.5v
tPHLAQ	25	NS	PROP. DELAY TIME, A TO Q FALLING	1.5	13	vdd=5v +/- 0.5v
tPHLWQ	25	NS	PROP. DELAY TIME, W TO Q FALLING	1.5	10	vdd=5v +/- 0.5v
tSDE	25	NS	SETUP TIME, D RELATIVE TO E	N/A	3.5	vdd=5v +/- 0.5v
tHDE	25	NS	HOLD TIME, D RELATIVE TO E	N/A	2	vdd=5v +/- 0.5v
tSAE	25	NS	SETUP TIME, A RELATIVE TO E	N/A	6	vdd=5v +/- 0.5v
tHAE	25	NS	HOLD TIME, A RELATIVE TO E	N/A	2	vdd=5v +/- 0.5v
tWMR	25	NS	MR PULSE WIDTH	N/A	6	vdd=5v +/- 0.5v
tWE	25	NS	E PULSE WIDTH	N/A	6	vdd=5v +/- 0.5v
PERIOD	1000	NS	PERIOD	50	N/A	

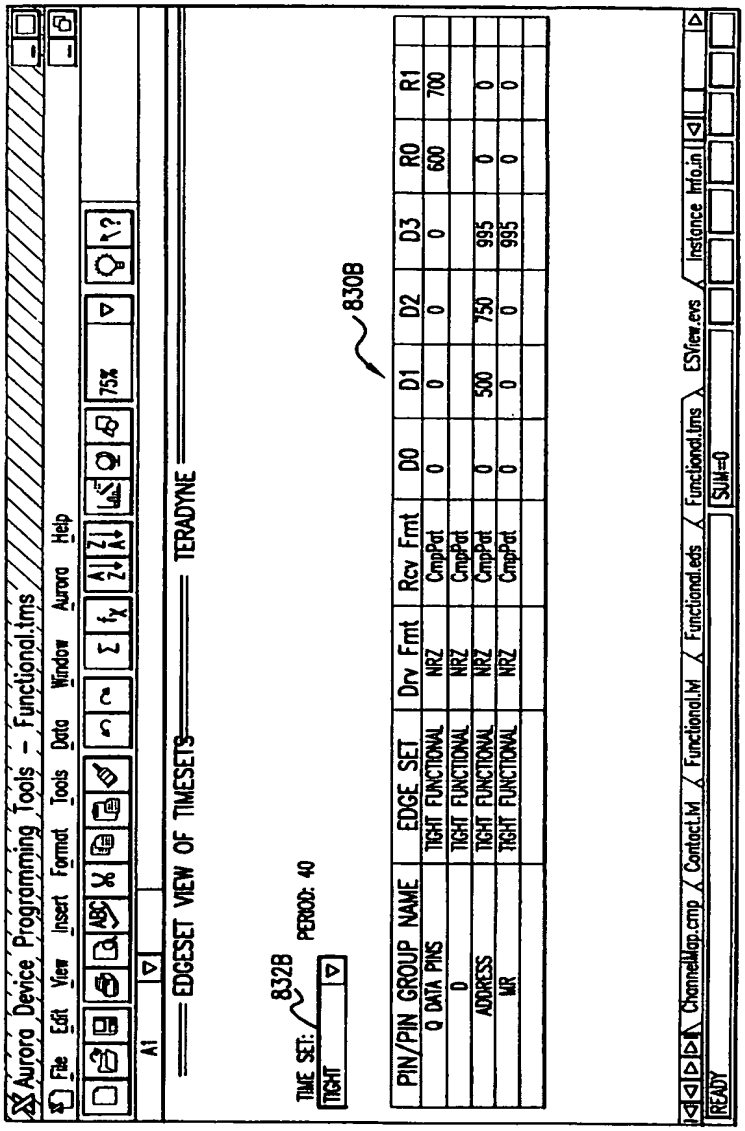
도면7


Aurora Device Programming Tools - Functional.eds
File Edit View Insert Format Tools Data Window Aurora Help
EDGESETS
TERADYNE
PIN NAME:
Q DATA PINS
TIMING MODE:
EXTENDED MODE
EDGE:
ALL USED
SHEET UNITS:
NANO SECS
PIN NAME | EDGE SET | DRV FMT | REV FMT | EDGE | VALUE | EQUATION | COMMENT
Q DATA PINS | LOOSE FUNCTIONAL | NRZ | CMPPAT | D0 | 0
D1 | 0
D2 | 0
D3 | 0
R0 | 600 | PERIOD/2 + 100
R1 | 700 | _DATA_PINS.LOOSE_FUNCTIONAL.R0 + 100
TIGHT FUNCTIONAL | NRZ | CMPPAT | D0 | 0
D1 | 0
D2 | 0
D3 | 0
R0 | 600 | PERIOD/2 + 100
R1 | 700 | _DATA_PINS.TIGHT_FUNCTIONAL.R0 + 100
Latch.pmp
Consumer.sps
ChannelMap.cmp
Contact.lvl
Functional.lvl
Functional.eds
Functional.tms
READY
SUM=0
730
740
750
752
754
756
758
760
762
764
766
768
770
772

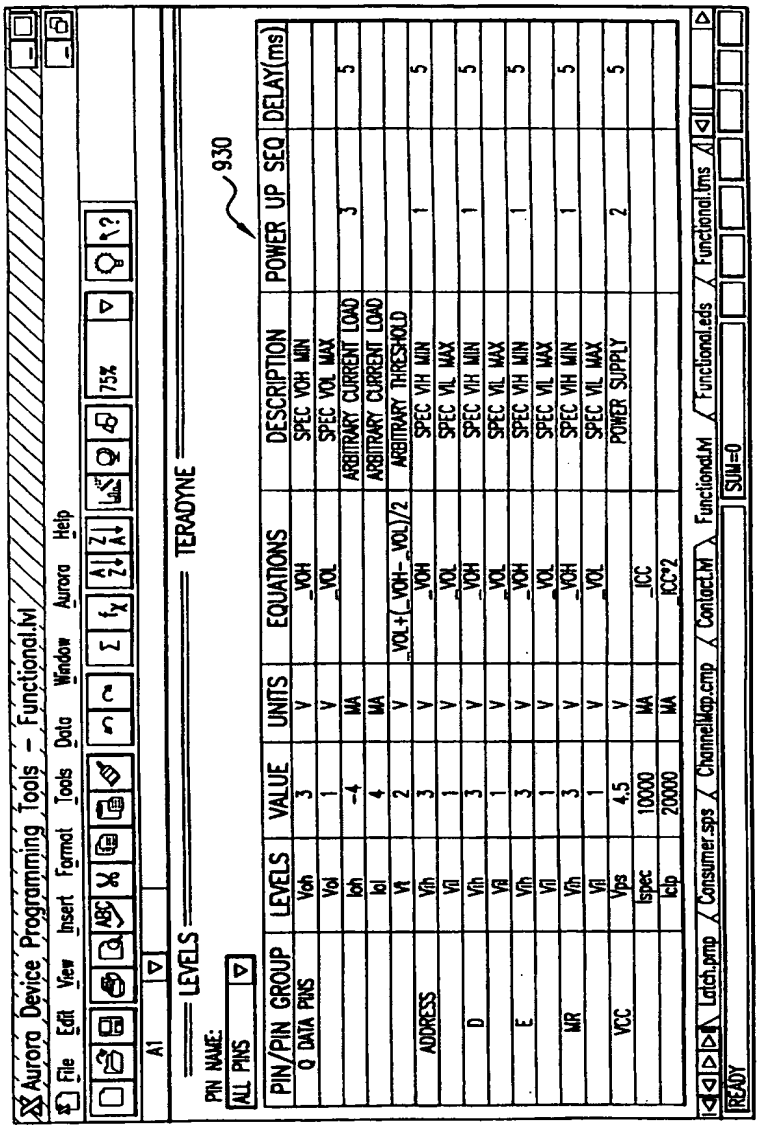
도면8a


Aurora Device Programming Tools - Functional.tms
File Edit View Insert Format Tools Data Window Aurora Help
TIME SETS
TERADYNE
EDGE SETS SHEET: FUNCTIONAL.eds
TIME SET:
832A
TIGHT
UNITS
834A
NANO SECS
830A
TIME SETS | PERIOD | EQUATIONS | PIN/PIN GROUP NAME | EDGE SETS
TIGHT | 40 | | Q DATA PINS | TIGHT FUNCTIONAL
| | | 0 | TIGHT FUNCTIONAL
| | | ADDRESS | TIGHT FUNCTIONAL
| | | MR | TIGHT FUNCTIONAL
Latch.pmp Consumer.sps ChannelMap.cmp Contact.M Functional.M Functional.eds Functional.tms
READY
SUM=0

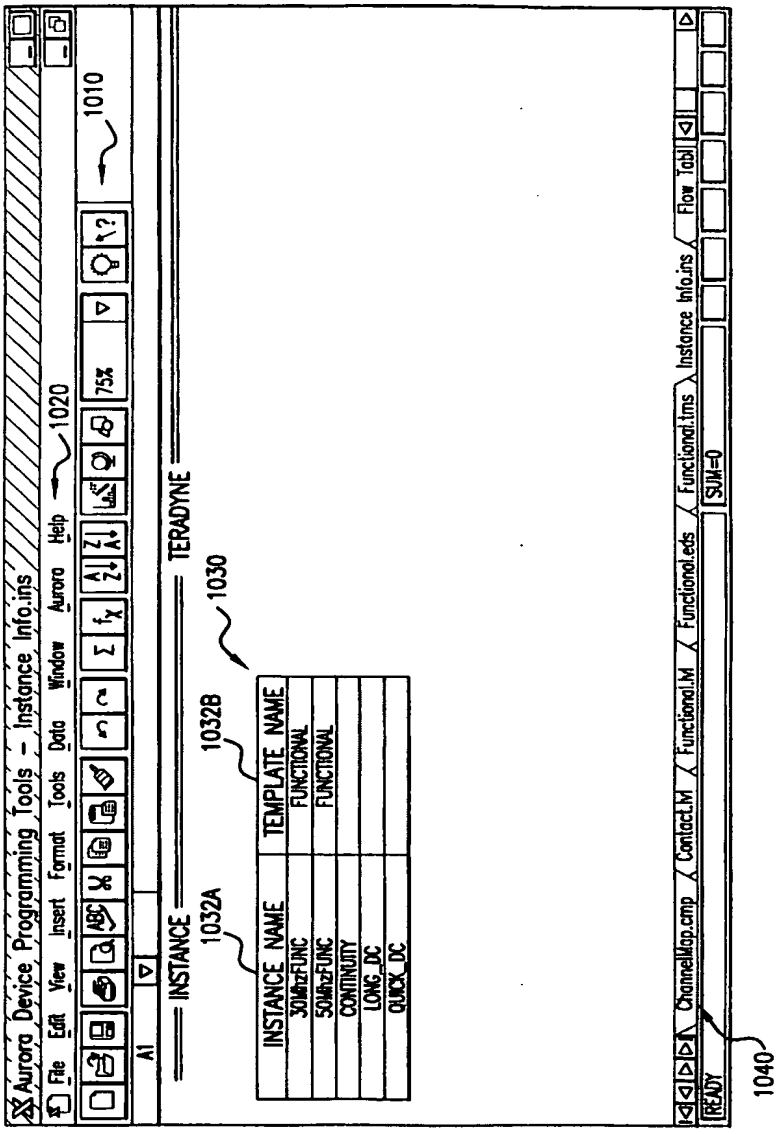
도면8b


Aurora Device Programming Tools - Functional.tms
File Edit View Insert Format Tools Data Window Aurora Help
EDGESET VIEW OF TIMESETS
TERADYNE
TIME SET:
TIGHT
PERIOD: 40
832B
830B
PIN/PIN GROUP NAME
EDGE SET
Drv Fmt
Rcv Fmt
D0
D1
D2
D3
R0
R1
Q DATA PINS
TIGHT FUNCTIONAL
NRZ
CmpPat
0
0
0
0
600
700
0
TIGHT FUNCTIONAL
NRZ
CmpPat
ADDRESS
TIGHT FUNCTIONAL
NRZ
CmpPat
0
500
750
995
0
0
MR
TIGHT FUNCTIONAL
NRZ
CmpPat
0
0
0
995
0
0
ChannelMap.cmp
Contact.lvl
Functional.lvl
Functional.eds
Functional.tms
ESView.evs
Instance Info.in
READY
SUM=0

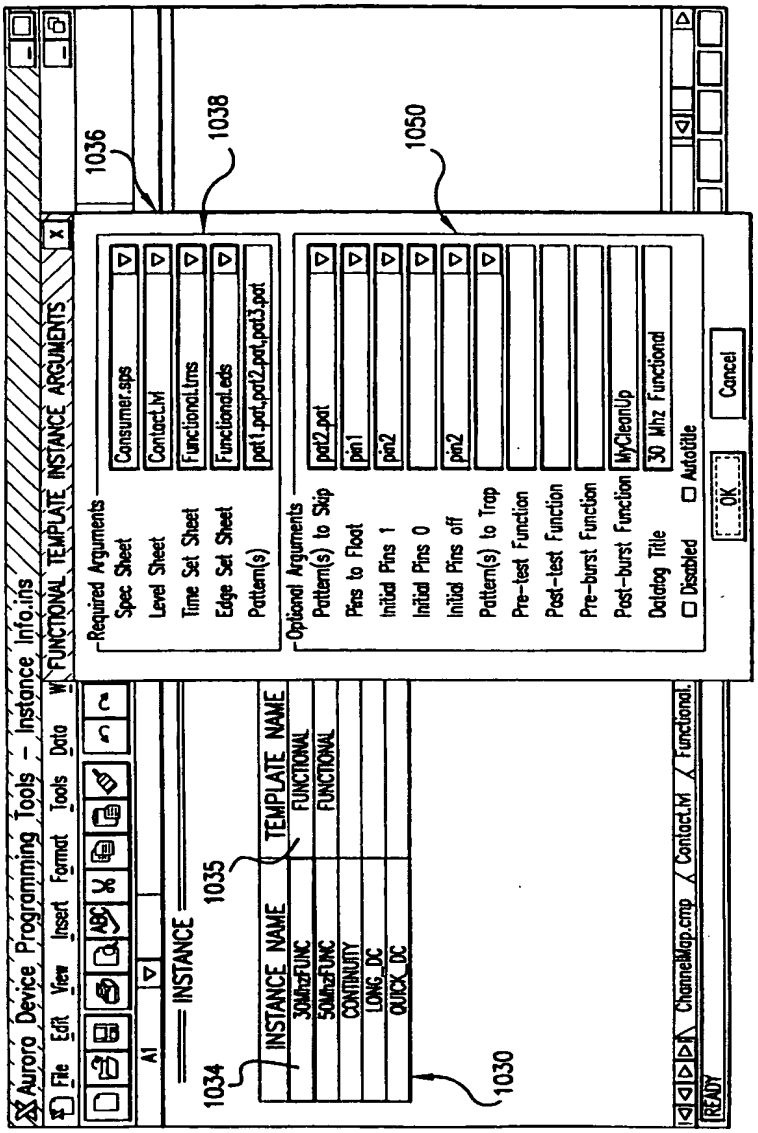
도면9


Aurora Device Programming Tools - Functional.lvl
File Edit View Insert Format Tools Data Window Aurora Help
A1
LEVELS
TERADYNE
930
PIN NAME:
ALL PINS
PIN/PIN GROUP | LEVELS | VALUE | UNITS | EQUATIONS | DESCRIPTION | POWER UP SEQ | DELAY(ms)
Q DATA PINS | Voh | 3 | V | _VOH | SPEC VOH MIN
Vol | 1 | V | _VOL | SPEC VOL MAX
Ioh | -4 | MA | ARBITRARY CURRENT LOAD | 3 | 5
Iol | 4 | MA | ARBITRARY CURRENT LOAD
Vt | 2 | V | _VOL+(_VOH-_VOL)/2 | ARBITRARY THRESHOLD
ADDRESS | Vih | 3 | V | _VOH | SPEC VIH MIN | 1 | 5
Vil | 1 | V | _VOL | SPEC VIL MAX
D | Vih | 3 | V | _VOH | SPEC VIH MIN | 1 | 5
Vil | 1 | V | _VOL | SPEC VIL MAX
E | Vih | 3 | V | _VOH | SPEC VIH MIN | 1 | 5
Vil | 1 | V | _VOL | SPEC VIL MAX
WR | Vih | 3 | V | _VOH | SPEC VIH MIN | 1 | 5
Vil | 1 | V | _VOL | SPEC VIL MAX
VCC | Vps | 4.5 | V | POWER SUPPLY | 2 | 5
Ispec | 10000 | MA | _ICC
Iclp | 20000 | MA | _ICC*2
Latch.pmp Consumer.sps ChannelMap.cmp Contact.lvl Functional.lvl Functional.eds Functional.tms
READY
SUM=0

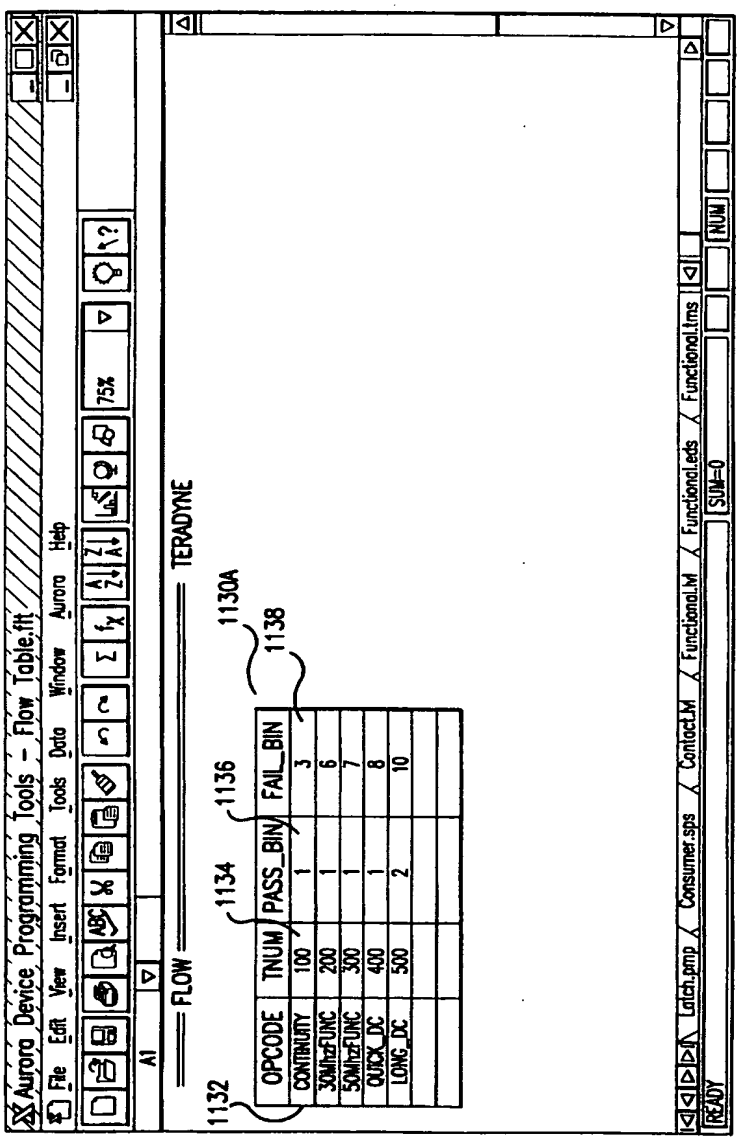
도면10a


Aurora Device Programming Tools - Instance Info.ins
File Edit View Insert Format Tools Data Window Aurora Help
1020
1010
A1
INSTANCE
TERADYNE
1032A
1032B
1030
INSTANCE NAME
TEMPLATE NAME
30MhzFUNC
FUNCTIONAL
50MhzFUNC
FUNCTIONAL
CONTINUITY
LONG_DC
QUICK_DC
ChannelMap.cmp
Contact.M
Functional.M
Functional.eds
Functional.tms
Instance Info.ins
Flow Tabl
SUM=0
READY
1040

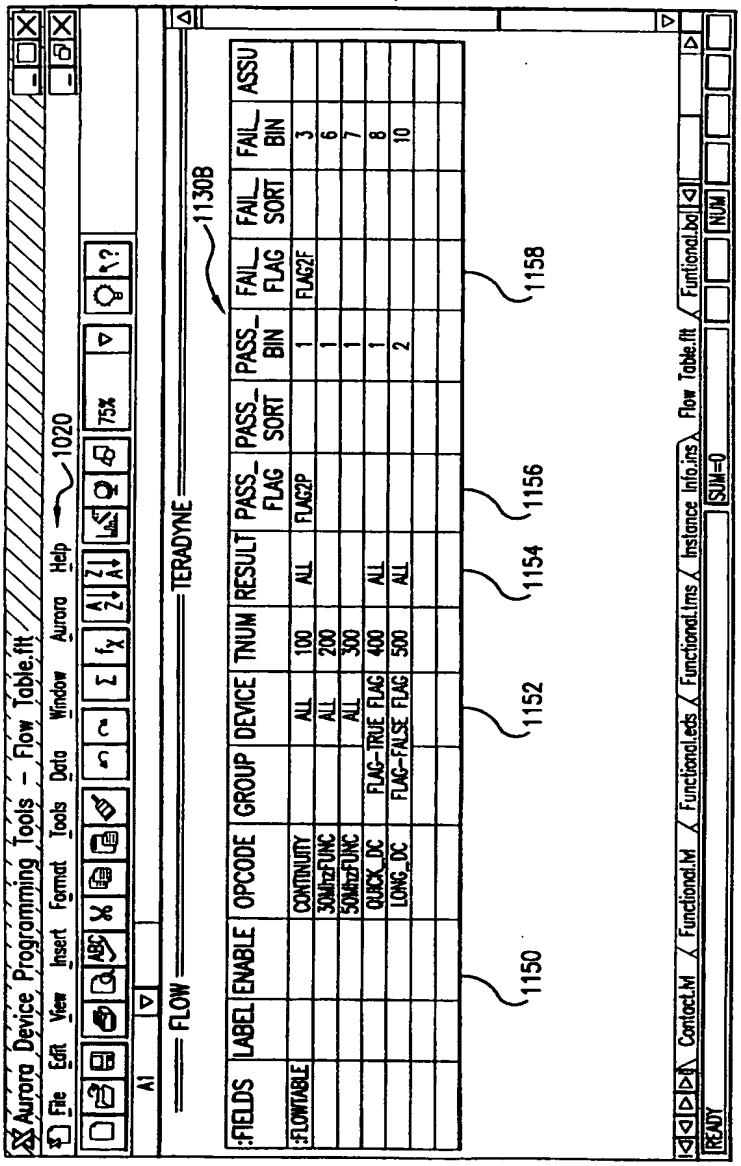
도면10b


Aurora Device Programming Tools - Instance Info.ins
File Edit View Insert Format Tools Data W FUNCTIONAL TEMPLATE INSTANCE ARGUMENTS
A1
INSTANCE
INSTANCE NAME
TEMPLATE NAME
30MhzFUNC
FUNCTIONAL
50MhzFUNC
FUNCTIONAL
CONTINUITY
LONG_DC
QUICK_DC
Required Arguments
Spec Sheet
Level Sheet
Time Set Sheet
Edge Set Sheet
Pattern(s)
Consumer.sps
Contact.lvl
Functional.tms
Functional.eds
pat1.pat,pat2.pat,pat3.pat
Optional Arguments
Pattern(s) to Skip
Pins to Float
Initial Pins 1
Initial Pins 0
Initial Pins off
Pattern(s) to Trap
Pre-test Function
Post-test Function
Pre-burst Function
Post-burst Function
Datalog Title
pat2.pat
pin1
pin2
pin2
MyCleanUp
30 Mhz Functional
Disabled
Autotitle
OK
Cancel
ChannelMap.cmp
Contact.lvl
Functional
READY
1036
1038
1050
1035
1034
1030

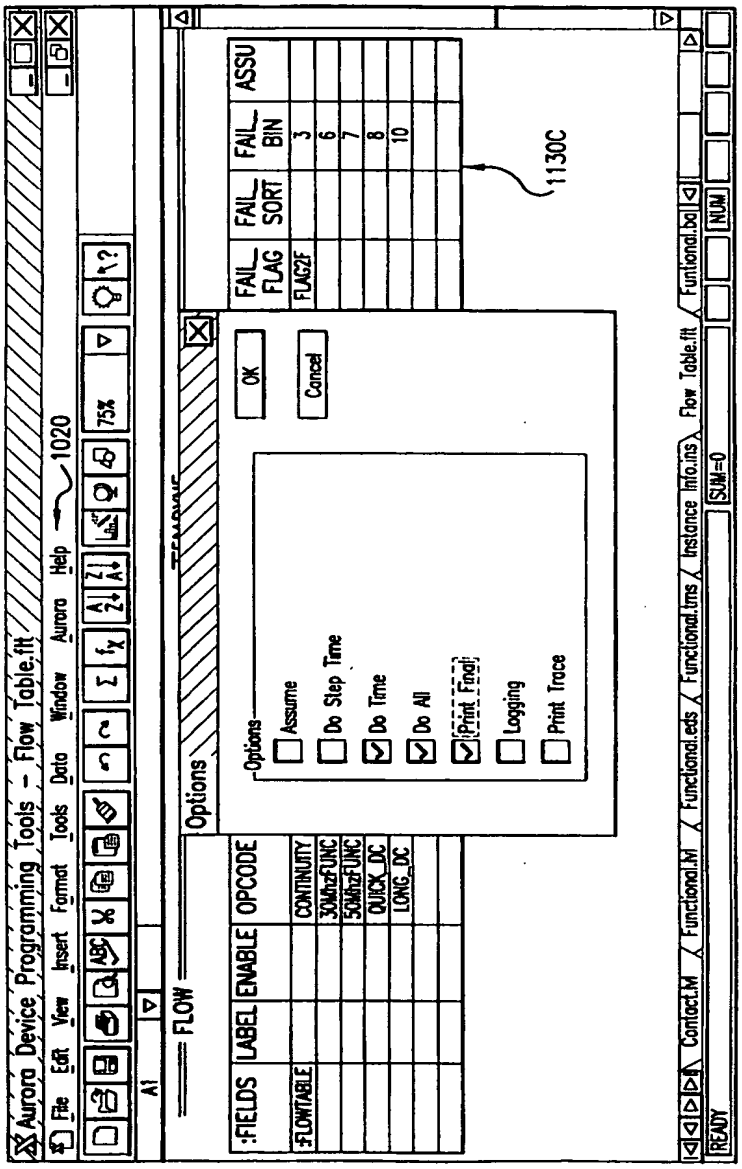
도면11a


Aurora Device Programming Tools - Flow Table.ftt
File Edit View Insert Format Tools Data Window Aurora Help
75%
A1
FLOW
TERADYNE
1132
1134
1136
1130A
1138
OPCODE
TNUM
PASS_BIN
FAIL_BIN
CONTINUITY 100 1 3
30MhzFUNC 200 1 6
50MhzFUNC 300 1 7
QUICK_DC 400 1 8
LONG_DC 500 2 10
Latch.pmp
Consumer.sps
Contact.M
Functional.M
Functional.eds
Functional.tms
READY
SUM=0
NUM

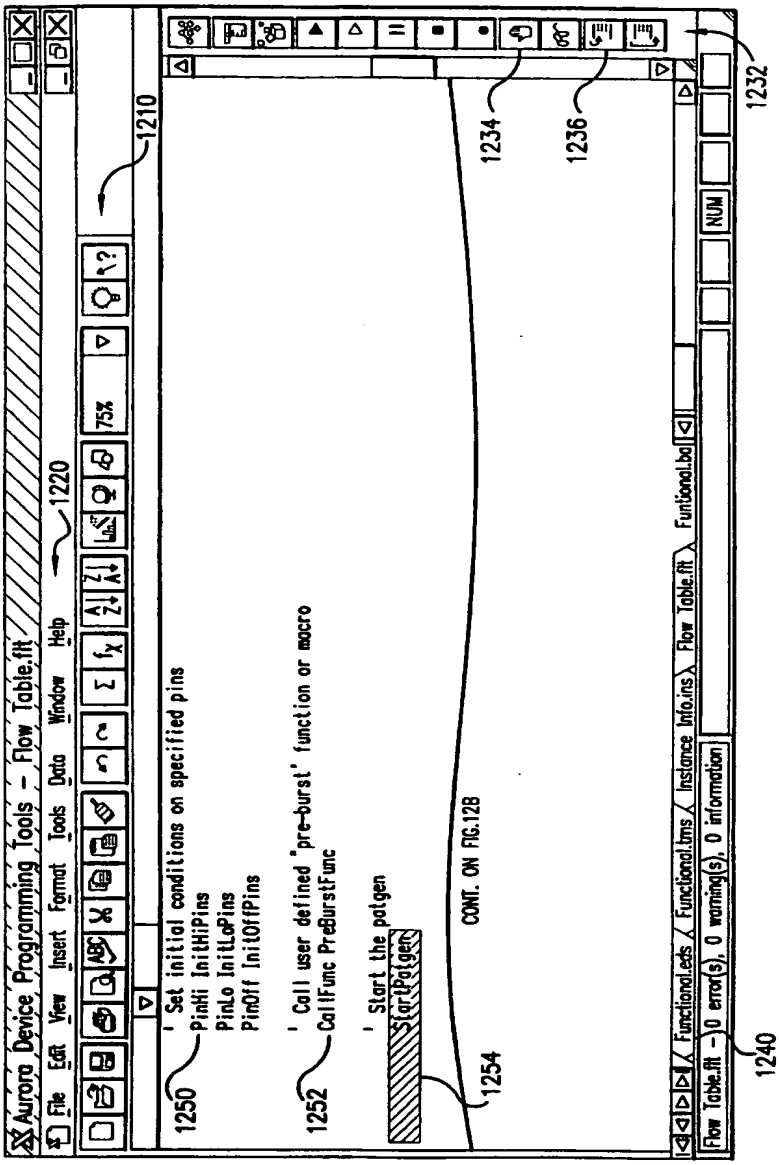
도면11b


Aurora Device Programming Tools - Flow Table.flt
File Edit View Insert Format Tools Data Window Aurora Help
1020
FLOW
TERADYNE
1130B
:FIELDS | LABEL | ENABLE | OPCODE | GROUP | DEVICE | TNUM | RESULT | PASS_FLAG | PASS_SORT | PASS_BIN | FAIL_FLAG | FAIL_SORT | FAIL_BIN | ASSU
:FLOWTABLE
CONTINUITY ALL 100 ALL FLAG2P 1 FLAG2F 3
50MhzFUNC ALL 200 1 6
50MhzFUNC ALL 300 1 7
QUICK_DC FLAG-TRUE FLAG 400 ALL 1 8
LONG_DC FLAG-FALSE FLAG 500 ALL 2 10
1150
1152
1154
1156
1158
Contact.M
Functional.M
Functional.eds
Functional.tms
Instance Info.ins
Flow Table.flt
Funtional.bol
READY
SUM=0
NUM

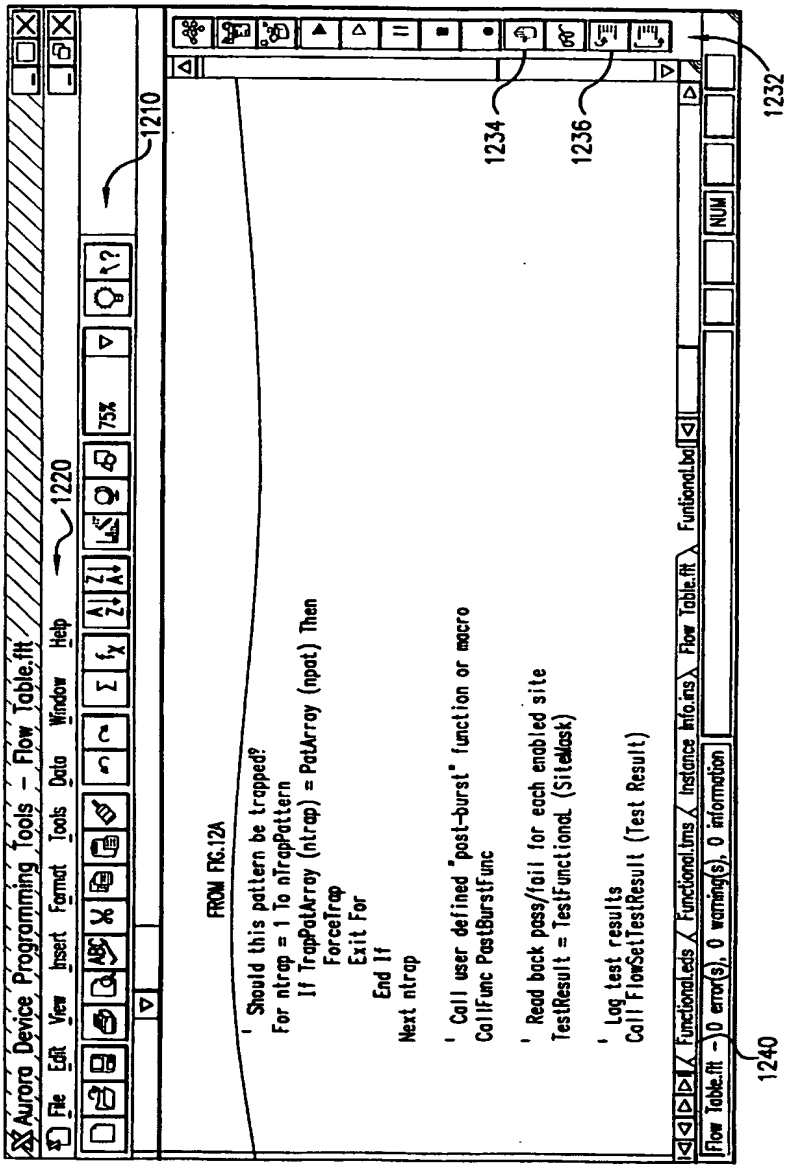
도면11c


Aurora Device Programming Tools - Flow Table.flt
File Edit View Insert Format Tools Data Window Aurora Help
1020
75%
A1
FLOW
Options
:FIELDS
LABEL
ENABLE
OPCODE
:FLOWTABLE
CONTINUITY
30MhzFUNC
50MhzFUNC
QUICK_DC
LONG_DC
OK
Cancel
Options
Assume
Do Step Time
Do Time
Do All
Print Final
Logging
Print Trace
FAIL_FLAG
FAIL_SORT
FAIL_BIN
ASSU
FLAG2F
3
6
7
8
10
1130C
Contact.M
Functional.M
Functional.eds
Functional.tms
Instance Info.ins
Flow Table.flt
Functional.bol
READY
SUM=0
NUM

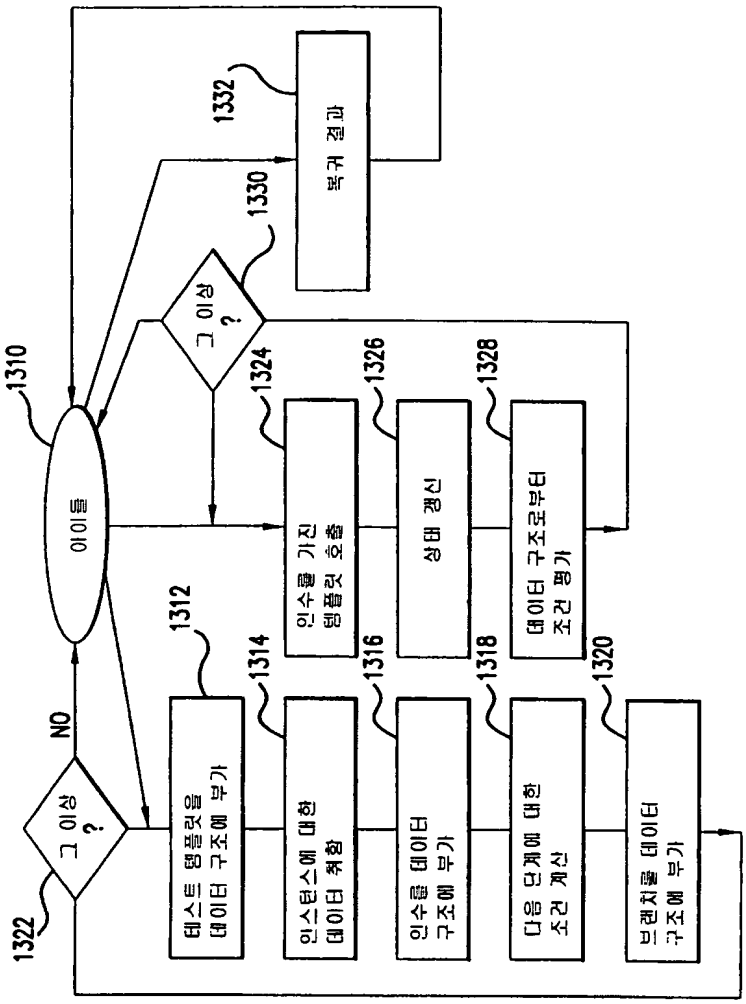
도면12a


Aurora Device Programming Tools - Flow Table.fit
File Edit View Insert Format Tools Data Window Help
1210
1220
1232
1234
1236
' Set initial conditions on specified pins
PinHi InitHiPins
PinLo InitLoPins
PinOff InitOffPins
1250
' Call user defined 'pre-burst' function or macro
CallFunc PreBurstFunc
1252
' Start the patgen
StartPatgen
1254
CONT. ON FIG.12B
Functional.eds
Functional.tms
Instance Info.ins
Flow Table.fit
Funtional.bd
Flow Table.fit - 0 error(s), 0 warning(s), 0 information
NUM
1240

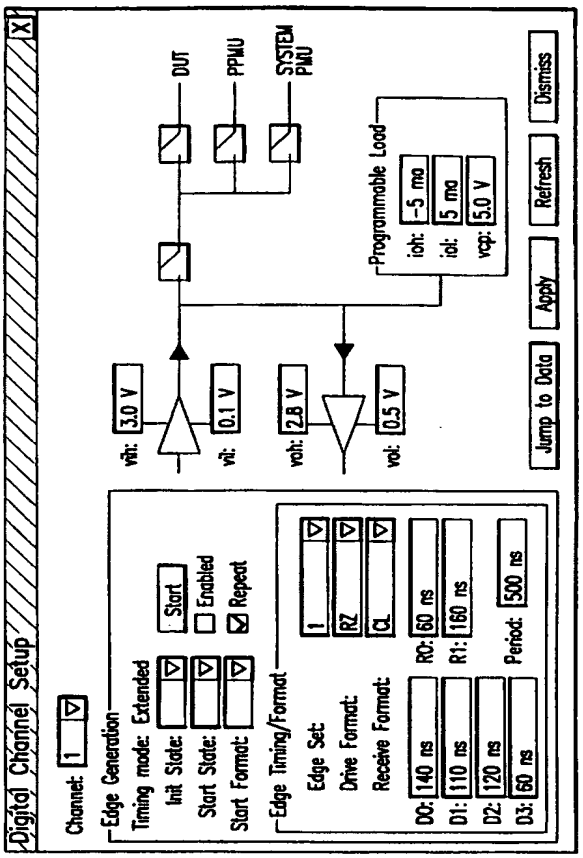
도면12b


Aurora Device Programming Tools - Flow Table.fit
File Edit View Insert Format Tools Data Window Help
1210
1220
1232
1234
1236
1240
FROM FIG.12A
' Should this pattern be trapped?
For ntrap = 1 To nTrapPattern
If TrapPatArray (ntrap) = PatArray (npat) Then
ForceTrap
Exit For
End If
Next ntrap
' Call user defined "post-burst" function or macro
CallFunc PostBurstFunc
' Read back pass/fail for each enabled site
TestResult = TestFunctional (SiteMask)
' Log test results
Call FlowSetTestResult (Test Result)
Functional.Leds
Functional.tms
Instance Info.ins
Flow Table.fit
Funtional.bxl
Flow Table.fit - 0 error(s), 0 warning(s), 0 information
NUM

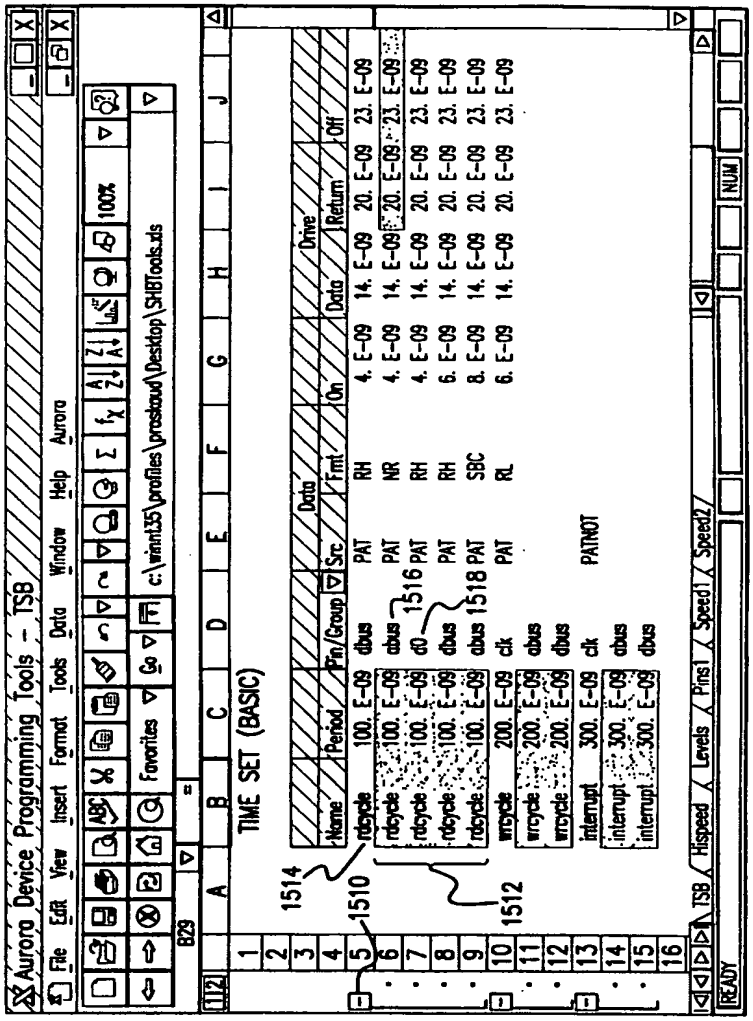
도면13


1310
이이들
1312
테스트 템플릿을
데이터 구조에 부가
1314
인스턴스에 대한
데이터 채움
1316
인수를 데이터
구조에 부가
1318
다음 단계에 대한
조건 계산
1320
브랜치를 데이터
구조에 부가
1322
그 이상
?
NO
1324
인수를 가진
템플릿 호출
1326
상태 갱신
1328
데이터 구조로부터
조건 평가
1330
그 이상
?
1332
복귀 결과

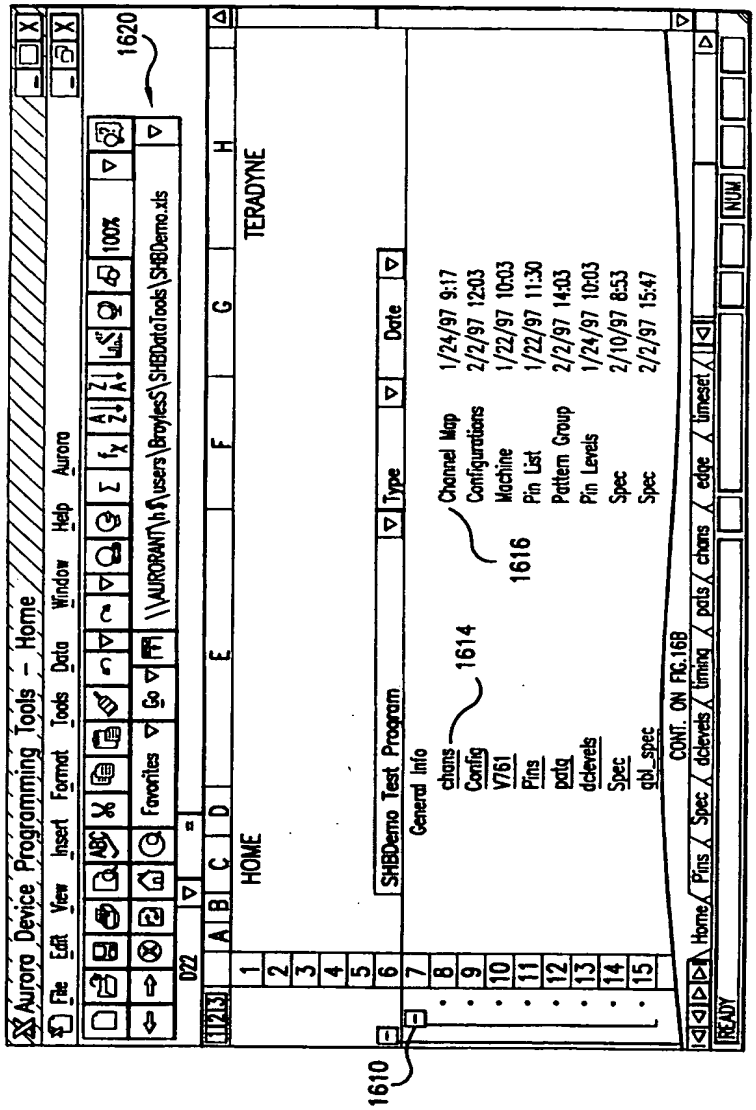
도면14


Digital Channel Setup
X1
Channel: 1
Edge Generation
Timing mode: Extended
Init State:
Start State:
Start Format:
Start
Enabled
Repeat
Edge Timing/Format
Edge Set:
Drive Format:
Receive Format:
1
RZ
CL
D0: 140 ns
D1: 110 ns
D2: 120 ns
D3: 60 ns
R0: 60 ns
R1: 160 ns
Period: 500 ns
voh: 3.0 V
vol: 0.1 V
voh: 2.8 V
vol: 0.5 V
DUT
PPMU
SYSTEM PMU
Programmable Load
ioh: -5 ma
iol: 5 ma
vcp: 5.0 V
Jump to Data
Apply
Refresh
Dismiss

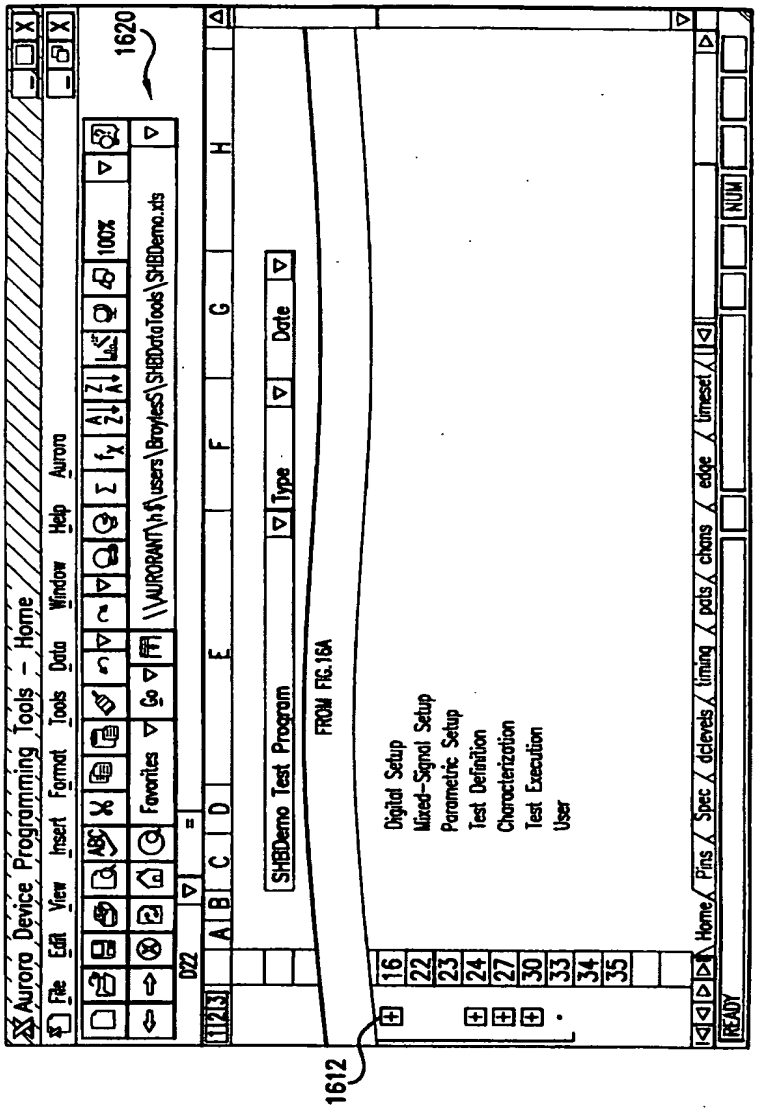
도면15


Aurora Device Programming Tools - TSB
File Edit View Insert Format Tools Data Window Help Aurora
TIME SET (BASIC)
1514
1510
1516
1518
1512
TSB Hispeed Levels Pins1 Speed1 Speed2
READY
NUM

도면16a


Aurora Device Programming Tools - Home
File Edit View Insert Format Tools Data Window Help Aurora
\\AURORANT\h$\Users\BroylesS\SHBDataTools\SHBDemo.xls
1620
HOME
TERADYNE
SHBDemo Test Program
General Info
chans
Config
V761
Pins
pdq
dclevels
Spec
gbl_spec
1614
Type
Date
Channel Map 1/24/97 9:17
Configurations 2/2/97 12:03
Machine 1/22/97 10:03
Pin List 1/22/97 11:30
Pattern Group 2/2/97 14:03
Pin Levels 1/24/97 10:03
Spec 2/10/97 8:53
Spec 2/2/97 15:47
1616
CONT. ON FIG.16B
Home Pins Spec dclevels timing pats chans edge timeset
READY
NUM
1610

도면16b


Aurora Device Programming Tools – Home
File Edit View Insert Format Tools Data Window Help Aurora
\\AURORANT\h$\Users\BroylesS\SHBDataTools\SHBDemo.xls
1620
SHBDemo Test Program
Type
Date
FROM FIG.16A
Digital Setup
Mixed-Signal Setup
Parametric Setup
Test Definition
Characterization
Test Execution
User
Home
Pins
Spec
dclevels
timing
pats
chans
edge
timeset
READY
NUM
1612